Universidad Nacional Mayor de San Marcos
Universidad del Perú. Decana de América
Facultad de Ingeniería Electrónica y Eléctrica
Escuela Profesional de Ingeniería de Telecomunicaciones

Propuesta de diseño de una red de datos de área local bajo la arquitectura de redes definidas por software para la Red Telemática de la Universidad Nacional Mayor de San Marcos

TESIS

Para optar el Título Profesional de Ingeniero de Telecomunicaciones

AUTOR

Joseph Daniel CHAFLOQUE MEJIA

ASESOR

Jesús Otto VILLANUEVA NAPURÍ

Lima, Perú

2018

Referencia bibliográfica

Chafloque, J. (2018). *Propuesta de diseño de una red de datos de área local bajo la arquitectura de redes definidas por software para la Red Telemática de la Universidad Nacional Mayor de San Marcos*. [Tesis de pregrado, Universidad Nacional Mayor de San Marcos, Facultad de Ingeniería Electrónica y Eléctrica, Escuela Profesional de Ingeniería de Telecomunicaciones]. Repositorio institucional Cybertesis UNMSM.

JOSEPH DANIEL CHAFLOQUE MEJIA

PROPUESTA DE DISEÑO DE UNA RED DE DATOS DE ÁREA LOCAL BAJO LA ARQUITECTURA DE REDES DEFINIDAS POR SOFTWARE PARA LA RED TELEMÁTICA DE LA UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS.

Tesis presentada a la Facultad de Ingeniería Electrónica y Eléctrica de la Universidad Nacional Mayor de San Marcos para obtener el Título de Ingeniero de Telecomunicaciones.

Área: Redes y Conectividad

Asesor: Jesús Otto Villanueva Napurí

Lima – Perú

2018

LISTA DE FIGURAS

LISTA DE TABLAS

RESUMEN

La presente tesis tiene como objetivo brindar una propuesta de diseño de una red de datos de área local bajo una arquitectura de redes definidas por software (SDN – Software Defined Network en sus siglas en ingles) para mejorar la eficiencia de la gestión e interoperabilidad entre los diferentes dispositivos o equipos de red que conforman la red de datos de área local (LAN – Lan Area Network en sus siglas en ingles) de la Red Telemática de la Universidad Nacional Mayor de San Marcos -UNMSM-.

Se explicará el funcionamiento de la arquitectura de redes definidas por software, los protocolos y plataformas que son utilizadas para su desarrollo. Se presenta un análisis de la forma de gestión de la red LAN tradicionales, como lo es la Red Telemática, y la arquitectura de los dispositivos de red tradicionales.

La propuesta del diseño de red se realizará de forma simulada bajo el software Mininet, se explicará la topología a diseñar, así como la descripción del controlador SDN a utilizar y finalmente se presentarán las pruebas y resultados obtenidos de la simulación.

Con los resultados obtenidos se comparará los beneficios que brinda la arquitectura SDN con respecto a la red LAN actualmente implementada, presentando las conclusiones y recomendaciones del proyecto de investigación.

ABSTRACT

The objective of this thesis is to provide a proposal for the design of a local area data network under an architecture of software defined networks (SDN - Software Defined Network in its acronym in English) to improve the efficiency of management and interoperability between different devices or network equipment that make up the local area data network (LAN - Lan Area Network in its acronym in English) of the Telematics Network of the National University of San Marcos -UNMSM-.

It will explain the operation of the architecture of networks defined by software, the protocols and platforms that are used for its development. An analysis of the management form of the traditional LAN network is presented, as is the Telematic Network, and the architecture of traditional network devices.

The proposed network design will be simulated under the Mininet software, the topology to be designed will be explained, as well as the description of the SDN controller to be used and finally the tests and results obtained from the simulation will be presented.

With the results obtained, the benefits provided by the SDN architecture will be compared with respect to the LAN network currently implemented, presenting the conclusions and recommendations of the research project.

ÍNDICE GENERAL

CAPITULO I

PLANTEAMIENTO DEL PROBLEMA

1.1. DEFINICIÓN DEL PROBLEMA

En los últimos años las redes de datos de área local bajo una arquitectura de red tradicional se han vuelto más complejas y robustas debido a los nuevos requerimientos de los servicios de redes, tales como Big Data, colaboración, voz sobre IP, Internet de las cosas, etc. En la mayoría de los casos, para cumplir con los nuevos servicios y/o aplicaciones, se usan diferentes dispositivos de red como switches, routers, firewall, entro otros., donde cada dispositivo de red cumple una función y consiguiente tienen configuraciones específicas, por lo cual mientras más compleja es la red, más difícil resulta gestionar la totalidad de los dispositivos instalados complicando la configuración de requerimientos de alto nivel (calidad de servicio, listas de acceso, entre otros). Adicionalmente la Open Networking Foundation (ONF,2012) identifico que red tradicional presenta los siguientes inconvenientes: red estática y compleja, políticas inconsistentes, falta de escalabilidad y dependencia de los proveedores de dispositivos.

La Red Telemática de la UNMSM mantiene una arquitectura de red de datos tradicional con las limitaciones mencionadas y no está preparada para los nuevos servicios (Big Data, streaming de audio y video, videoconferencias, etc) y tecnologías de redes actuales (Internet de las cosas, quinta generación de comunicaciones moviles, etc). La arquitectura de las Redes Definidas por Software, en ingles Software Defined Networking (SDN), supera esta situación problemática ya que, mediante el uso de protocolos abiertos, optimiza la gestión de los dispositivos de red con un único centro de control para todos ellos (switch, routers, firewall) independientemente de la marca del fabricante. SDN es una arquitectura de red flexible y automatizada abierta a nuevas posibilidades futuras.

Es por esto por lo que resulta relevante analizar la alternativa de un diseño de red de datos de área local bajo la arquitectura de una red definida por software para la Red Telemática y dar un alcance para futuros estudios e implementaciones sobre este tema.

1.2. JUSTIFICACIÓN

En la arquitectura de una red de datos de área local tradicional, la gestión (implementación de nuevos requerimientos de red) requiere personal técnico altamente calificado en la configuración de diferentes dispositivos de red, donde la interacción entre estos dispositivos (routers, switches, etc.) es compleja, adicionalmente las redes actuales cuentan con dispositivo de diferentes fabricantes (Cisco, Juniper, Fortinet, etc.) haciendo que el costo del mantenimiento de la red sea alto. La arquitectura de red que busca mitigar estos inconvenientes, que se propone en esta investigación, es la de *"redes definidas por software"* (software-defined networking - SDN).

Con la realización de la presente investigación, brindaremos una propuesta de nuevo diseño de red a la Red Telemática. La nueva arquitectura de red tendría un control centralizado que permitirá una visibilidad completa de la red, logrando la optimización de las operaciones y una mayor flexibilidad de la gestión de los dispositivos, permitiéndonos agilizar los tiempos de configuraciones de la red independientemente del fabricante, ya que no sería necesario conocer la forma de configuración de cada fabricante ya que tendremos un controlador centralizado el cual comunicará las ordenes bajo un protocolo en común llamado Openflow.

La importancia de esta investigación consiste en evaluar la flexibilidad de le gestión de los dispositivos de red LAN empleando las bondades de SDN y la utilización de software libre, dotando a la Red Telemática de la UNMSM con una red de datos programable y adaptable.

1.3. ANTECEDENTES

Actualmente no existe una única definición para la arquitectura de redes definidas por software, sin embargo, el concepto de SDN puede ser definido como una red definida por un software que permite separar el plano de control (software) del plano de datos (hardware) con un único centro de gestión de los dispositivos flexible, programable y con mayor visibilidad de la red.

El término Software Defined Networking se ha dado a conocer en los recientes años, sin embargo, el concepto original nació en 1996, derivado del deseo de proporcionar un controlador para administración total del reenvió de paquetes en los nodos de una red de los proveedores de servicios.

Los primeros trabajos de los grupos de investigación, pero no limitadas a las siguientes propuestas, que contribuyeron a la evolución de esta nueva arquitectura de red fueron:

- **NEWMAN et al. (1996)**, participantes del grupo de trabajo de la Internet Engineering Task Force (IETF), presentaron la RFC1987 en donde se propone el protocolo General Switch Management protocol (GSMP) que permite tener un control centralizado de los switches en redes ATM, facilitando la operabilidad y mantenimiento de las redes.

- **ROONEY; VAN DER MERWET (1998)** publicaron el artículo "The Tempset: A framework for safe, resource-assured, programable networks", dando a conocer la posibilidad de tener redes programables de forma segura y con diferentes funciones de red personalizadas.

- **YANG ed al. (2004),** del grupo de trabajo de la IETF, presentaron el estándar RFC3746, separando los planos de control (elemento de control) y el plano de datos (reenvió de paquetes) en los nodos de la red, pero interconectándolos mediante una interfaz estándar de comunicación llamada ForCES.

- **LUO ed al. (2007)** brindo una propuesta de una nueva arquitectura de administración de seguridad de switches basados en flujo con un controlador central que administra el enrutamiento de estos flujos, llamado Ethane.

- La Open Networking Foundation (ONF) estandarizo el protocolo Openflow, el cual está definido en los documentos OpenFlow Switch Specification, la versión 1.1.0 del protocolo fue lanzada en el 2011, no obstante, el desarrollo del protocolo empezó en el 2008 en la Universidad de Stanford. El protocolo Openflow es el protocolo que más se ha adaptado a las soluciones de SDN ya que permite establecer entradas en un flujo de tablas y exportarlas a un controlador centralizado.

Según Ignasio Errando, "*SDN sustituye el nivel de control del hardware de red por una capa de software abstraída mediante técnicas de virtualización, tratando de hacer la red más programable. Esto se concreta en distintas aproximaciones, como son: proporcionar un acceso al hardware basado en programación mediante protocolos como OpenFlow; arquitecturas construidas sobre un nivel de control basado en software; y crear redes virtuales por encima del hardware que dirijan el tráfico a través de redes físicas"* (Data Center Dinamics, 2003, p 26-27).

Este comentario acerca del uso protocolos de código abierto, como Openflow, enfatiza la creación de "redes virtuales" que permitan el control sobre la red independientemente del fabricante del dispositivo de red.

En los últimos 4 años se han desarrollado investigaciones sobre diseños e implementaciones experimentales que utilizan este nuevo concepto de arquitectura de red SDN, tales como:

- **DANDEKAR; KHARADE (2015)** en su artículo "Implementing a low cost SDN ecosystem in LAN", concluyeron que un switch bajo la arquitectura SDN puede comportarse como cualquier dispositivo de red, inclusive el administrador de la red tendría la posibilidad de diseñar flujos personalizados, logrando más flexibilidad y facilidad en la operación y administración de los dispositivos de red.

- **CUBA; BECERRA (2015)** en su tesis "Diseño e implementación de un controlador sdn/openflow para una red de campus académica", realizaron la implementaron de prueba de un controlador basado en una solución Floodlight escalable en para el tráfico unicast, mediante el mecanismo usado por el módulo Clustering, además demostraron que usando este mecanismo se obtiene un 25% de uso en las TCAM de los Switches y en la capacidad del controlador logrando optimizar el tráfico de la Red Campus.

- **N. ESPAÑA (2016)** presentó el trabajo de investigación "Diseño y simulación de una red definida por software (SDN)" donde destacó en esta nueva estructura de red que, independientemente del lenguaje de programación que utilicen los controladores, se puede migrar los módulos ya establecidos al nuevo entorno de red SDN.

En el presente trabajo se tienen en cuenta las investigaciones expuestas y se buscará utilizar las recientes versiones de los controladores disponibles para uso de investigación.

1.4. OBJETIVOS

1.4.1. OBJETIVO GENERAL

Proponer un diseño de una red LAN bajo la arquitectura SDN para la Red Telemática de la UNMSM en un entorno de simulación, que permita optimizar la gestión e interoperabilidad de los dispositivos de red mediante un control centralizado.

1.4.2. OBJETIVOS ESPECÍFICOS

- Analizar el estado de la Red Telemática de la UNMSM según la forma de la gestión de los dispositivos de red en redes de arquitectura tradicionales.
- Diseñar la red LAN de la Red Telemática de la UNMSM bajo una arquitectura de red SDN mediante el programa de simulación Mininet buscando optimizar la gestión de los dispositivos de red.
- Evaluar la gestión de los dispositivos de red de la arquitectura SDN propuesta para la red LAN de La Red Telemática bajo el entorno simulado.

1.5. METODOLOGÍA DE TRABAJO

Para esta tesis con característica de propuesta de diseño de red, se tendrá una metodología basada en tres etapas: documentación, evaluación de la información recolectada y diseño.

- Documentación

En la construcción de esta tesis es necesario contar con informaciones relevantes de los diseños de redes bajo un entorno de redes definidas por software (SDN), además de evaluar las diferentes formas de simulación sobre Mininet, así como los diferentes controladores SDN y conocer las formas de gestión de redes tradicionales como la Red Telemática de la UNMSM. Para obtener esta información se usarán distintas fuentes, herramientas y estrategias entra las cuales se tienen:

- Consultar diferentes fuentes de información en el internet, así como bibliotecas físicas y virtuales.
- Investigación de los dispositivos de red existentes, controladores SDN y formas de virtualización.

- Evaluación de la información obtenida

Una vez obtenida toda la información requerida, se realizará un adecuamiento de la topología de red de la Red Telemática con el fin de realizar la simulación cercana en el software Mininet.

- Propuesta de diseño - Experimental

La propuesta de diseño de red implicara simular bajo software libre, en tanto el software lo permita, los dispositivos de red teniendo en cuenta un único centro de control donde se logre gestionar centralizadamente todos los dispositivos de red simulados, tomando en cuenta la viabilidad desde el punto de vista tecnológico, características y requerimientos.

Se plasmarán todos los resultados obtenidos a fin de evaluar el desempeño de la propuesta de diseño.

1.6.ORGANIZACIÓN DE LA TESIS

La tesis está estructura de la siguiente manera.

- En el primer capítulo se brinda una introducción general donde se indica la problemática de la Tesis, los objetivos de la tesis, la justificación y antecedentes de la tesis.
- El segundo capítulo presenta el marco teórico del concepto de la "*arquitectura de redes definidas por software*" (SDN), sus diferentes protocolos y funcionamiento.
- En el tercer capítulo trata de la forma de gestión de redes tradicionales y la topología de la Red Telemática adaptada del año 2016.
- En el cuarto capítulo se realiza el planteamiento del diseño y la simulación de una red telemática bajo la arquitectura SDN.
- En el quinto capítulo se realiza las pruebas y evaluación del diseño presentado utilizando la simulación lograda.
- En el sexto se describe los beneficios económicos de SDN.
- En el séptimo capitulo se brinda las líneas de investigación que nos proporciona SDN.
- En el octavo, noveno y décimo capítulo se presentan las conclusiones, recomendaciones y observaciones.

CAPITULO II

MARCO TEÓRICO CONCEPTUAL

2.1. NUEVAS TENDENCIAS EN LA RED

Las nuevas tecnologías en la red están demandando mayores recursos de red y de administración ya que estas consumen mayor velocidad de transmisión y requieren mayor número de dispositivos de red para mantenerlo.

Según **CISCO (2017)**, el tráfico de internet está en aumento y se estima que en el año 2021 el tráfico de internet llegara a 278 exabytes por mes, estos incrementos del tráfico de internet hacen reevaluar los enfoques de la arquitectura de red tradicional.

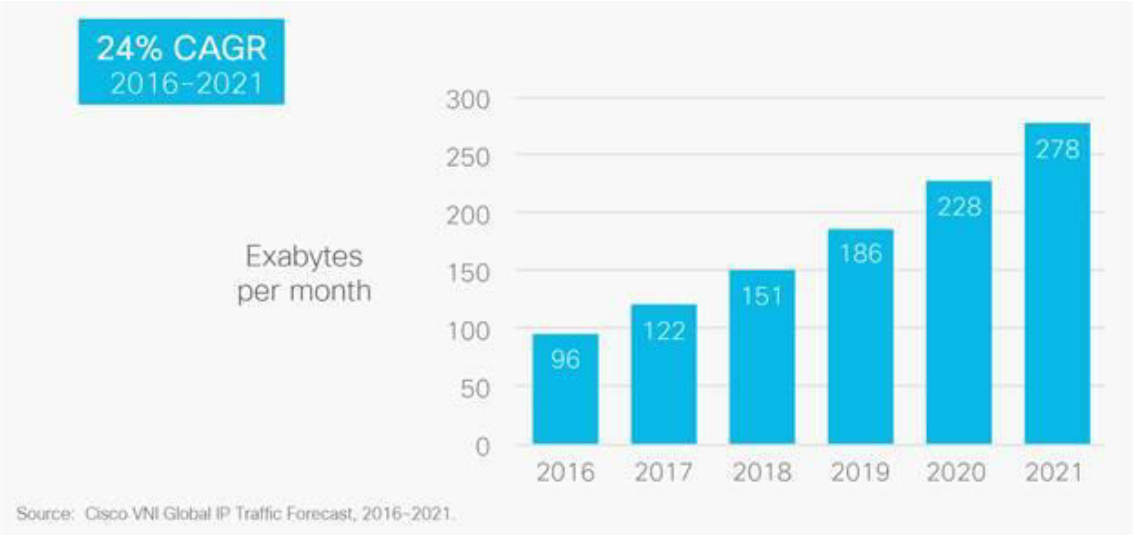

Figura 2.1: Crecimiento del tráfico IP global.
Fuente: CISCO VNI (2016)

STALLINGS (2015) indica que las nuevas tendencias en la red pueden agruparse en computación en la nube, big data, tráfico móvil e internet de las cosas.

2.1.1. COMPUTACIÓN EN LA NUBE

Este concepto de computación en la nube está en aumento en el mundo y presenta nuevos desafíos en las redes actuales ya que requiere un eficiente flujo de datos a través de las redes de internet.

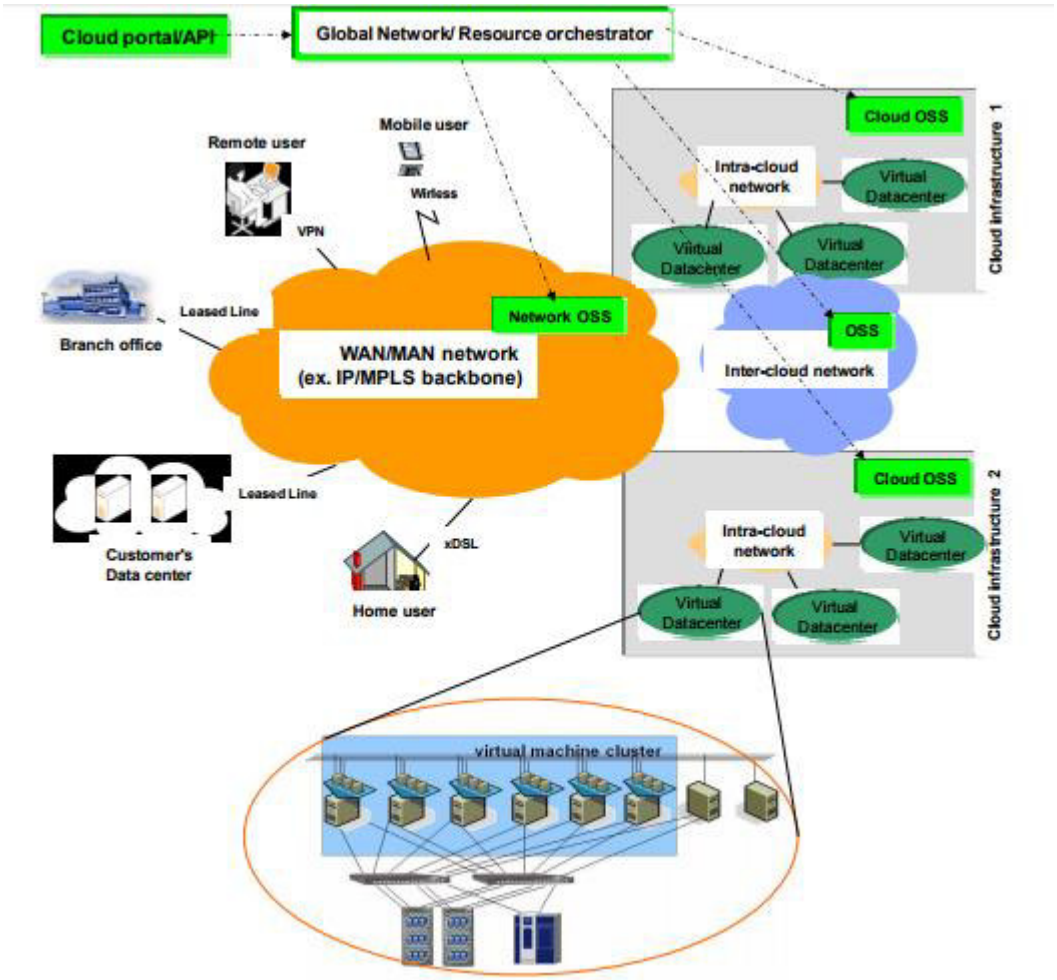

Figura 2.2: Modelo de red en la nube.
Fuente: ITU-T (2012)

La **ITU-T (2012)** desarrollo un modelo de arquitectura de red en la nube (Figura 2.2), en donde se aprecia que una red en la nube puede conectar diferentes dispositivos de una red de datos tradicional, las cuales pueden incluir servidores de bases de datos, firewalls, balanceadores de carga, IDS / IPS, etc. La red en la nube probablemente incluirá una serie de LAN interconectadas, incluso se podrá instalar un controlador SDN en la nube. La infraestructura de red en la nube contara con servidores de base de datos disponibles como un conjunto de máquinas virtuales, proporcionando entornos virtuales aislados para diferentes usuarios.

2.1.2. BIG DATA

El término Big Data se refiere a toda actividad relacionada que una organización utiliza para crear, manipular y administrar un conjunto muy grande de datos (en terabytes, petabytes, exabytes, etc.) así como las instalaciones en las que se almacenan, llamados centros de datos de almacenamiento físicos o en la nube.

Big Data se una tecnología demandante en estos tiempos ya que los bytes de datos continúan creciendo y cada vez son más debido a que son recolectados por numerosos sensores, dispositivos móviles, cámaras, micrófonos, lectores de identificación por radiofrecuencia (RFID), internet de las cosas y otras tecnologías similares. **BHAMBRI (2012**) estimó que diariamente se crean 2.5 exabytes (2.5 × 1018 bytes) de datos. Este gran aumento de datos requiere un procesamiento masivo en paralelo en miles de servidores, los cuales se interconectan entre sí ocasionando una gran demanda de la capacidad de la red.

2.1.3. TRAFICO MÓVIL

Actualmente, la tecnología ha contribuido a la transformación de lo que originalmente eran sólo teléfonos móviles en teléfonos inteligentes con nuevas características, como acceso a Internet, aplicaciones móviles, cámaras digitales de alta calidad, acceso a múltiples tipos de redes inalámbricas (Wi-Fi, Bluetooth, 3G, 4G, etc.).

Por este motivo, los usuarios acceden cada vez más a los recursos de la red empresarial a través de sus dispositivos móviles, sean teléfonos inteligentes, tablets y/o laptops. Estos dispositivos admiten sofisticadas aplicaciones que pueden consumir y generar tráfico de imagen y video, incrementando la carga de tráfico en la red empresarial conllevando a tener la necesidad de añadir más dispositivos de red para soporten el aumento de tráfico.

2.1.4. INTERNET DE LAS COSAS

El Internet de las cosas (IoT) es un término que se refiere a la interconexión de los dispositivos inteligentes, que van desde los aparatos cotidianos a sensores pequeños. Esta interconexión entre los sensores de corto alcance instalados en aparatos cotidianos permite nuevas formas de comunicación entre las personas y las cosas, y entre las cosas en sí. El Internet apoya ahora la interconexión de billones de objetos industriales y personales, generalmente a través de los sistemas de la nube. Los diferentes objetos monitoreados entregan información de los sensores, en algunos casos logran actuar sobre su entorno para modificar y crear una gestión general de un sistema más grande, como una fábrica o incluso una ciudad.

La mayoría de las "cosas" en el Internet de las cosas genera un tráfico modesto, aunque hay excepciones, como las cámaras de video de vigilancia. Pero el gran número de dispositivos de este tipo, para algunas empresas, resulta en una carga significativa para la red empresarial.

2.2.LIMITACIONES DE LA ARQUITECTURA DE RED TRADICIONAL

Las redes actuales son el resultado de una serie de protocolos y arquitecturas de red que se desarrollaron inicialmente en los años 1970, donde la primera red de computadoras ARPANET hacia su aparición. En ese momento, se preveía que 32 bits del campo del direccionamiento IP serían suficientes para manejar todas las direcciones de Internet Protocol (IPv4) que Internet necesitaría (aproximadamente 16 millones) sin embargo, el

3 de febrero de 2011, la IANA asignó los últimos bloques libres a los Registro Regional de Internet (RIRs) agotando el pool de direcciones IPv4 disponibles. Una vez establecida, la arquitectura de red no ha cambiado mucho en las últimas décadas, sin embargo, los servidores que se conectan a las redes de internet actuales han sufrido una dramática transformación en la última década.

Con la llegada de la virtualización, los servidores han cambiado su funcionalidad, los servidores son ahora dinámicos ya que es posible crear y/o mover fácilmente un servidor virtual, derivando en un aumento del número de servidores que pueden utilizar la red. Antes, las aplicaciones se asociaban con un único servidor que tenía una ubicación fija en la red, con la ayuda de la virtualización, las aplicaciones se pueden distribuir a través de varias máquinas virtuales (VM), cada una de las VM pueden intercambiar flujos de tráfico con los demás. Los administradores de red pueden mover las máquinas virtuales para optimizar y reequilibrar las cargas de trabajo del servidor. El movimiento de la aplicación puede hacer que cambien los puntos finales físicos de un flujo existente. La capacidad de migrar VMs crea desafíos para muchos aspectos de la llamada red tradicional.

Junto con la virtualización de servidores, muchas compañías también están utilizando una sola red para entregar todas las necesidades de red de voz, video y datos que tienen, teniendo la necesidad de proporcionar un nivel diferenciado de servicio para diferentes aplicaciones, lo cual es conocido como calidad de servicio (QoS). En la arquitectura de red tradicional, el aprovisionamiento de muchas herramientas QoS en la mayoría de los casos se realiza de manera manual. El administrador de la red debe configurar el dispositivo de cada proveedor por separado y ajustar parámetros como el ancho de banda de la red y la calidad de servicio en cada sesión por aplicación, esto demanda tener el conocimiento de la configuración de cada dispositivo por cada proveedor que posea.

Basado en estas nuevas tendencias, la **OPEN NETWORKING FOUNDATION (ONF, 2012)** ha identificado 4 limitaciones generales de la arquitectura de red tradicional:

- **Arquitectura estática y compleja:** Para responder a demandas como:
    - Niveles diferentes de QoS, aumento de la capacidad de la red soportada y políticas de seguridad más complejas. La red empresarial se ha vuelto más compleja y difícil de gestionar.
    - Nuevas demandas de la red se han desarrollado una serie de protocolos definidos independientemente, donde cada uno se encarga de una parte de los requisitos de red. Un ejemplo de la dificultad que esto presenta es cuando se agregan nuevos dispositivos.

Debido a estos inconvenientes, el administrador de la red se ve obligado a utilizar herramientas de administración a nivel de dispositivo para realizar cambios en los parámetros de configuración en varios dispositivos de red, sean switches, routers, firewalls, etc.

Las actualizaciones incluyen cambios en listas de control de acceso (ACL), configuración de VLAN, configuración de QoS en numerosos dispositivos y otros ajustes relacionados con el protocolo. Otro ejemplo es el ajuste de los parámetros de QoS para satisfacer las necesidades cambiantes de los usuarios y los patrones de tráfico. Los procedimientos

manuales se deben utilizar para configurar el equipo de cada proveedor por cada servicio o aplicación e incluso por sesión.

Según la **OPEN DATA CENTER ALLIANCE (ODCA,2014),** las principales limitaciones se relacionan con:

- **Políticas inconsistentes:** Para implementar una política de seguridad en toda la red, en muchos casos, se tiene que realizar cambios de configuración en varios dispositivos y mecanismos. Por ejemplo, en una red grande, cuando se activa una nueva máquina virtual, puede tardar horas o incluso días en reconfigurar las ACL en toda la red.

- **Falta de escalabilidad:** Las demandas en las redes están creciendo rápidamente, tanto en tráfico como en el tipo de tráfico. La adición de más switches y/o capacidades de transmisión, que involucra múltiples equipos de proveedores, es difícil debido a la naturaleza estática de la red. Una estrategia que las empresas están utilizado es sobrescribir los enlaces de red basados en los patrones de tráfico previstos, sin embargo, con el creciente uso de la virtualización y la creciente variedad de aplicaciones multimedia, los patrones de tráfico son impredecibles.

- **Dependencia de los proveedores:** Dada la naturaleza de las demandas actuales de tráfico en las redes, las empresas y los operadores necesitan desplegar nuevos servicios y aumentar la capacidad de la red rápidamente en respuesta a las necesidades cambiantes de los negocios y las demandas de los usuarios. Pero la falta de aplicaciones abiertas para las funciones de red deja a las empresas limitadas a los ciclos de productos relativamente lentos de los proveedores de dispositivos de red.

2.3.NECESIDADES DE LAS NUEVAS REDES ACTUALES

Considerando las limitaciones de las redes actuales, listadas por la ONF, la **OPEN DATA CENTER ALLIANCE (ODCA,2014),** establece una lista útil y concisa de soluciones, que incluyen lo siguiente.

- **Adaptabilidad:** Las redes deben adaptarse y responder dinámicamente, basándose en las necesidades de las aplicaciones, las políticas empresariales y las condiciones de la red.

- **Automatización**: Los cambios de política deben propagarse automáticamente para reducir el trabajo manual y los errores.

- **Mantenibilidad:** La introducción de nuevas características y capacidades (actualizaciones de software, parches) debe ser perfecta con un mínimo de interrupción de las operaciones.

- **Gestión de modelos:** El software de gestión de redes debe permitir la gestión de la red a nivel de modelo, en lugar de implementar cambios conceptuales mediante la reconfiguración de elementos de red individuales.

- **Movilidad:** La funcionalidad de control debe adaptarse a la movilidad, incluidos los dispositivos de usuario móviles y los servidores virtuales.

- **Seguridad integrada:** las aplicaciones de red deben integrar la seguridad continua como un servicio central en lugar de como una solución complementaria.

- **Escalamiento de la red:** las implementaciones deben tener la capacidad de escalar o reducir la red y sus servicios para dar soporte a solicitudes bajo demanda.

Según la "Open Data Center Alliance" ODCA, la arquitectura de red definidas por software tiene la capacidad de satisfacer las necesidades mencionadas.

2.4. COMPONENTES DE UN DISPOSITIVO DE RED TRADICIONAL

Previo a la descripción de la arquitectura de red SDN, se brindará una explicación de cómo funciona un dispositivo dentro de una red tradicional para comprender el enfoque de SDN.

Según **KREUTZ (2015),** la funcionalidad de los dispositivos de red tradicionales se puede clasificar en tres planos, de datos, de control y de gestión (Figura 2.3).

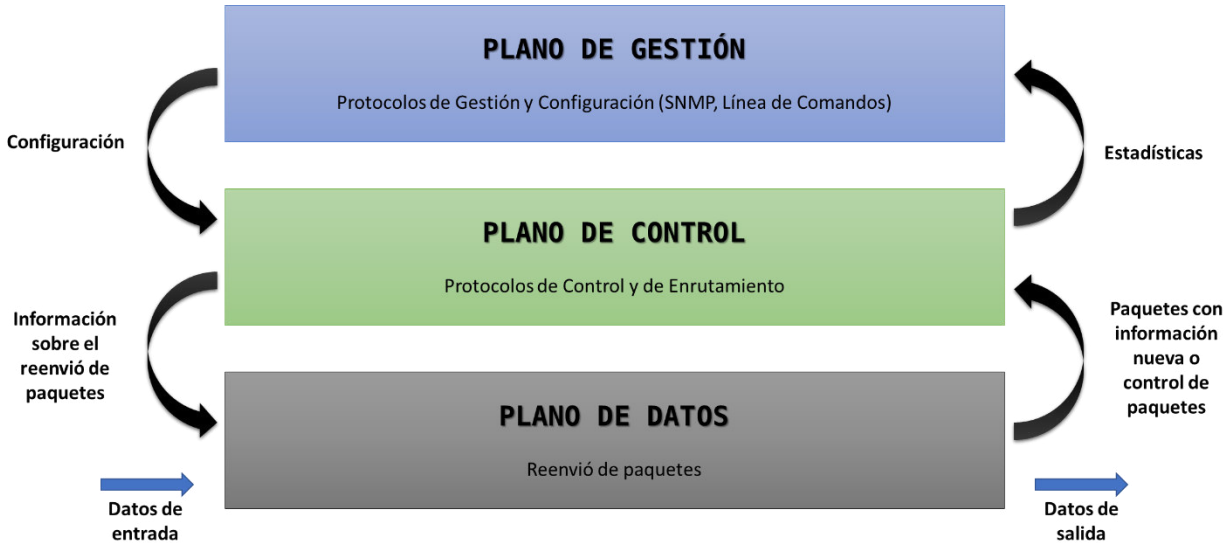

Figura 2.3: Planos de un dispositivo tradicional.
Fuente: KREUTZ (2015)

Cada plano realiza funciones específicas y pueden comunicarse entre ellas en el mismo dispositivo. Los tres planos se detallan a continuación:

2.4.1. PLANO DE CONTROL

El plano de control es el nivel más alto ya establece el conjunto de datos local que será utilizado para crear la tabla de reenvío de paquetes que, a su vez, son utilizadas por el plano de datos para reenviar tráfico entre puertos de entrada y salida en el dispositivo.

Según **NADEAU; GRAY (2013),** el conjunto de datos utilizado para almacenar la topología de la red es la tabla de enrutamiento, inglés *routing information base* (RIB). El RIB generalmente se mantiene consistente, a través del intercambio de información entre otras instancias de planos de control de otros dispositivos dentro de la red. Las entradas de la tabla de reenvío son denominadas comúnmente como la base de información de reenvío, o en sus siglas en inglés forwarding information base (FIB). Las tablas RIBs y FIBs reflejan la separación entre los planos de control y de datos de un dispositivo típico (Figura 2.4).

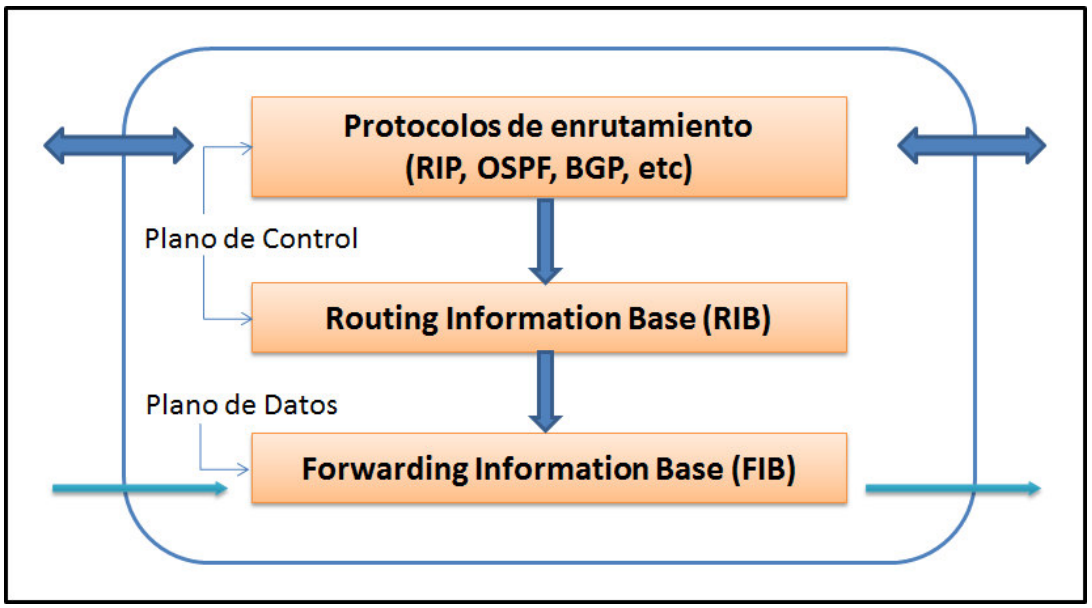

Figura 2.4: Tablas RIB y FIB.
Fuente: NADEAU; GRAY (2013)

El FIB se programa una vez que el RIB se considera consistente y estable, para mantener estable la tabla RIB, el plano de control tiene que desarrollar una vista de la topología de la red. Esta visión de la red puede ser programada manualmente, aprendida a través de la observación, o construida a partir de la información recopilada a través de otras instancias de planos de control, que puede ser mediante el uso de uno o varios protocolos de enrutamiento (por ejemplo, RIP, OSPF, EIGRP, IS-IS, etc.), programación manual o una combinación de ambos.

Según **NADEAU; GRAY (2013),** la mecánica de los planos de control y de datos se muestra en la figura 2.5, que representa una topología de red básica, con routers interconectados. En la parte superior de la figura se muestra dos router interconectados, y en la parte inferior se muestra una ampliación de los detalles de los planos de control y de datos de estos routers (indicados como A y B). En la figura, los paquetes son recibidos por el Router A por el lado izquierdo y finalmente, reenviados al conmutador B en el lado derecho de la figura. Dentro de cada dispositivo, los planos de control y de datos están separados, con el plano de control ejecutándose en su propio

procesador y/o tarjeta y el plano de datos en otro procesador y/o tarjeta independiente, ambos contenidos en un solo chasis.

De acuerdo con **NADEAU; GRAY (2013),** en la figura 2.5, los paquetes se reciben en los puertos de entrada del Router A en la tarjeta de línea donde reside el plano de datos. Si, por ejemplo, se recibe un paquete con destino de una IP nuevo o desconocido para la tabla FIB, el router redirige la información de este paquete (2) a su plano de control, donde se aprende, procesa y después se envía hacia su siguiente destino (3). Este mismo tratamiento se da para controlar el tráfico como mensajes de protocolo de enrutamiento (por ejemplo, anuncios de estado de enlace OSPF, "Open Shortest Path First).

Una vez que un paquete ha sido entregado al plano de control, la información contenida en el paquete es procesada y posiblemente resulta en una modificación de la tabla RIB, así como la transmisión de mensajes adicionales a los demás routers, alertándolos de una nueva actualización (es decir, una nueva ruta es aprendida). Cuando la tabla RIB se estabiliza, la tabla FIB se actualiza tanto en el plano de control como en el plano de datos. Posteriormente, el reenvío se actualizará y reflejará estos cambios para que cuando otro paquete llegue quiero llegar al mismo destino IP, solo sea procesado en el plano de datos y reenviar de acuerdo con las entradas de la tabla FIB. Cabe decir que todo cambio en la tabla RIB, sea manual o automático (mediante protocolos internos), la tabla FIB será actualizada. El mismo algoritmo para el procesamiento de paquetes sucede en el router de la derecha.

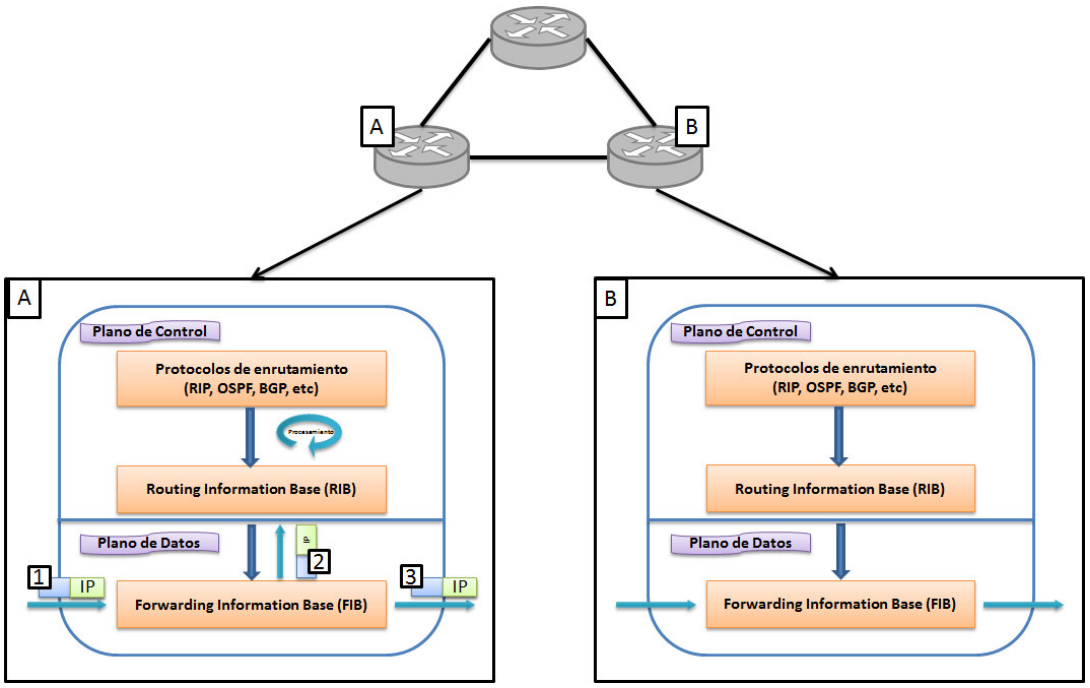

Figura 2.5: Topología de red tradicional.
Fuente: NADEAU; GRAY (2013)

Según **NADEAU; GRAY (2013)**, debido a las nuevas tendencias de la red, los protocolos de plano de control que buscan optimizar rutas optimizadas se enfrentan a varios desafíos. Esto incluye un crecimiento creciente de la base de información utilizada, es decir, el

crecimiento del tamaño de la tabla de enrutamiento, y cómo administrarlo. Esto, a su vez, puede conducir a altas tasas de tráfico solo para cambios en la red.

La misma evolución y problemática tienen lugar tanto para las redes de capa 2 como de capa 3 y en los protocolos que conformaron el plano de control. Un plano de control de capa 2 se enfoca en direcciones de hardware o de capa física, como direcciones IEEE MAC. Un plano de control de capa 3 utiliza direcciones de la capa de red como las del protocolo IP.

Las redes de capa 2 se ocupan principalmente del almacenamiento de direcciones MAC con fines de reenvío. Dado que las direcciones MAC de los hosts pueden ser enormes en una red de grandes empresas, la gestión de estas direcciones es complicada, complicándose aún más cuando se intenta gestionar todas las direcciones MAC en varias empresas. Los mecanismos aprendizaje de las direcciones MAC y los mecanismos utilizados para garantizar un esquema redundante (por ejemplo, el protocolo "Spanning Tree" (STP), también tiene limitaciones de escalabilidad.

Según **NADEAU; GRAY (2013)**, las redes de capa 2 no escalan bien debido al gran número de host finales derivados del aumento de las tecnologías inalámbricas y la masificación del internet de las cosas (IOT). Sin embargo, existen estándares de protocolos de control de capa 2 cuyos objetivos eran mitigar estos inconvenientes, entre ellos tenemos el protocolo Transparent Interconnection of Lots of Links (TRILL) de la IETF y el protocolo 802.1aq Shortest Path Bridging (SPB) de la IEEE, pocos usadas en las redes empresariales debido a su complejidad, no obstante la arquitectura SDN busca también mitigar estos inconvenientes con las llamadas tablas de flujo, el cual permitirá un mayor control del QoS de los paquetes.

2.4.2. PLANO DE DATOS

De acuerdo con **NADEAU; GRAY (2013)**, el plano de datos reenvía las tramas de datos recibidas en la interfaz física del dispositivo (sea de cable UTP, fibra óptica, medios inalámbricos, etc.) a través de una serie de operaciones de nivel de la capa de enlace del modelo OSI. Recogen la trama en su buffer, realizan las modificaciones en la cabecera para reenviar la trama. La trama de datos se procesa en el plano de datos realizando búsquedas del destino en la tabla FIB, que son programadas previamente por el plano de control. Este procedimiento se denomina el reenvió rápido de paquetes porque solo necesita identificar el destino de la trama usando la tabla FIB previamente configurada. La única excepción a este procedimiento es cuando la trama no coincide con ninguna entrada de la tabla FIB (por ejemplo, cuando se tiene un destino desconocido por la tabla FIB), en estos casos el paquete es enviado al plano de control, donde el paquete es procesado en la tabla RIB.

Según **NADEAU; GRAY (2013)**, el plano de datos también es asociado al hardware, ya que la velocidad de conmutación de paquetes está impulsada por el hardware del dispositivo (GPU/CPU). Las búsquedas en las tablas FIB basadas en el hardware resultan tener un rendimiento de reenvío de paquetes mucho mayor y por lo tanto han dominado los diseños de los dispositivos de red, particularmente para los dispositivos de red que

utilizan un ancho de banda más alto, ya que estos utilizan tarjetas de línea dedicada al reenvió de paquetes.

Según **KREUTZ (2015),** las acciones típicas resultantes de la búsqueda de reenvío en el plano de datos son las de conmutación del paquete, descarte del paquete, el conteo, etc. Algunas de estas acciones pueden ser combinadas o encadenadas juntas. En algunos casos, los paquetes están destinado a un proceso que se ejecuta localmente, como los protocolos de enrutamiento (OSPF, BGP, etc.), estos paquetes son reenviados directamente al plano de control.

Además de la decisión de reenvío, el plano de datos puede implementar otros servicios y/o funciones relacionadas el reenvió de paquetes como las listas de acceso (ACL) o la calidad de servicio (QoS). Dependiendo del diseño del dispositivo, se pueden implementar diferentes características dependiendo del fabricante del dispositivo, en algunos casos puede que algunas funciones sean exclusivas de otros fabricantes.

Según **GORANSSON (2016)**, el procedimiento local utilizada para la búsqueda de reenvío puede ser alterado, por ejemplo, una política de QoS puede asignar un flujo a una cola al salir o entrar para normalizar el servicio con las políticas a través de la red. Y, al igual que una ACL, puede descartar algunos paquetes independientemente de las entradas en la tabla FIB existentes.

2.4.3. PLANO DE GESTIÓN

Según **GORANSSON (2016)**, el plano de gestión es responsable de coordinar la interacción entre el plano de control y el plano de datos. Los mecanismos comunes para gestionar los dispositivos de red incluyen la interfaz de línea de comandos (CLI), el protocolo SNMP, HTTP, etc. Los administradores de red son los usuarios finales que utilizan el plano de gestión para la interacción con el dispositivo final.

Según **NADEAU; GRAY (2013)**, el plano de control y de datos mencionados tiene protocolos estándares que garantizan la interoperabilidad básica entre los dispositivos de red, el plano de gestión puede implementarse con estándares completamente propietarios. Protocolos como SNMP son útiles, pero para una mejor experiencia de usuario la mayoría de los proveedores implementar su propio CLI y / o Element Management Systems (EMS) donde la sintaxis de configuración de la gestión de red no es interoperable entre dispositivos de red de diferentes fabricantes, por ejemplo, CLI para Cisco IOS no es compatible con Juniper JUNOS CLI. Esto hace que la gestión de las redes de varios proveedores sea muy engorrosa.

2.5. REDES DEFINIDAS POR SOFTWARE (SDN)

El principal concepto que aborda la arquitectura SDN es permitir a los desarrolladores y administradores de redes tener el mismo tipo de control sobre el dispositivo de red que el que se tiene actualmente sobre los servidores para lograr afrontar muchos de los problemas que enfrentan los administradores de las redes tradicionales.

Según la **ODCA (2014),** el enfoque SDN divide la función del plano de datos y un plano de control en dispositivos separados (Figura 2.6). El plano de datos es solo responsable del reenvío de paquetes, mientras que el plano de control proporciona la "inteligencia" en el diseño de rutas, estableciendo parámetros de prioridad y de direccionamiento para cumplir con los requisitos de QoS y para hacer frente a los requerimientos de la red. Las interfaces abiertas se definen de modo que el hardware de conmutación presente una interfaz uniforme independientemente del proveedor, de igual forma, se definen interfaces abiertas para permitir que las aplicaciones de red se comuniquen con los controladores SDN. El controlador centralizado tiene el conocimiento de todos los componentes de red, teniendo la capacidad de gestionar a los switches mediante mensajes de software.

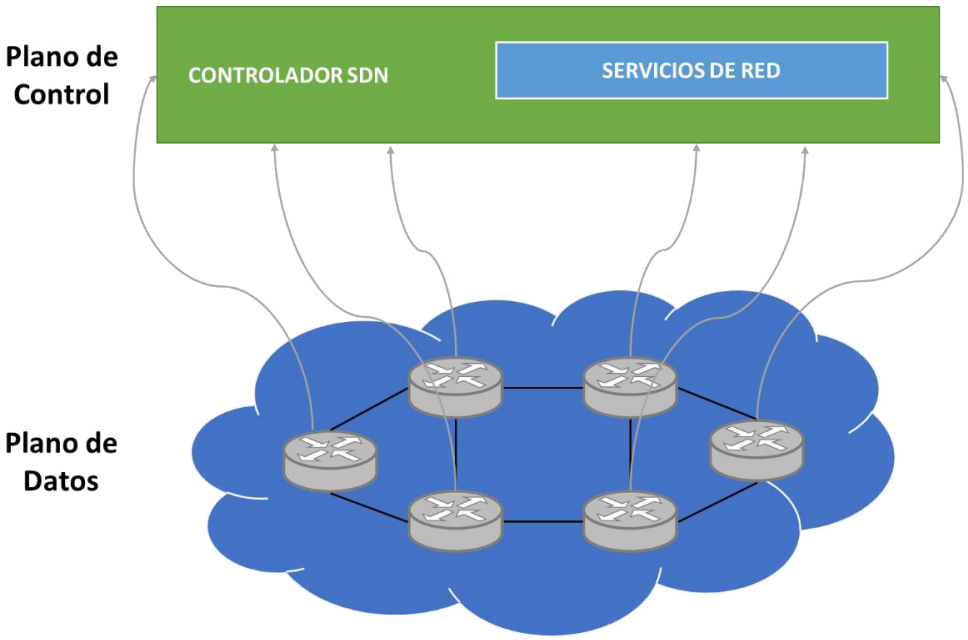

Figura 2.6: Concepto de una red definida por software.
Fuente: ODCA (2014)

Según la **ODCA (2014)**, los principios fundamentales de la arquitectura SDN son la separación de los planes de red, gestión centralizada y protocolos estandarizados. Cada uno de ellos se describen a continuación:

- **Separación de los planos de red**: Con la separación de los planos, se puede realizar nuevos modelos de gestión de la red que ofrezcan flexibilidad, mayor control y nuevos modelos de servicio innovadores. Específicamente, la separación de capas permite la programación de la red a través de interfaces de programación de aplicaciones, o en sus siglas en ingles API (Application Programing Interfaz), para de esta manera aprovechar el procesamiento distribuido y facilitar la implementación modular de aplicaciones de red para los planos de gestión y control.

    - Aplicaciones de red: Las nuevas plataformas de software que optimizan las funciones de control, servicio y gestión de capas pueden conducir a

nuevos marcos de gestión y soluciones empresariales. Este es el área de mayor promesa en SDN.

- **Gestión centralizada:** Las operaciones de la red pueden simplificarse centralizando muchas de las funciones de la administración, los servicios de red y los planos de control. El control de redes autónomas (como sucursales, campus, data center) puede ser lógicamente centralizado sin afectar la capacidad de comunicación entre redes ni de reducir la autonomía de esta, no obstante, puede ayudar a resolver muchos de los problemas actuales de la gestión de la red.

- **Protocolos estandarizados:** La adopción de protocolos estandarizados ayuda a lograr la interoperabilidad de múltiples proveedores permitiendo la elección para reducir costos.

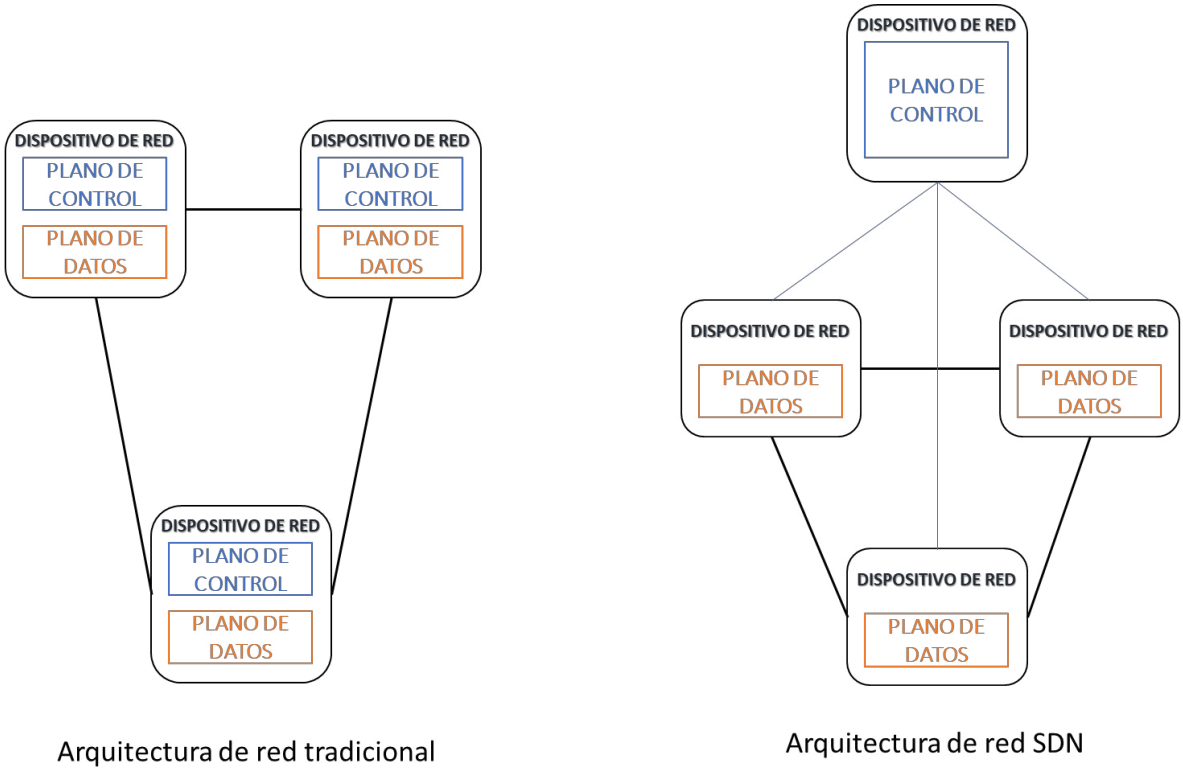

Figura 2.7: Red tradicional y red definida por software.
Fuente: Elaboración propia

2.6. ARQUITECTURA DE LAS REDES DEFINIDAS POR SOFTWARE (SDN)

HALEPLIDIS ed al. (2015), grupo de trabajo de la IETF, emitieron la RFC 7426. En el documento de la RFC7426 se brinda los lineamentos de la arquitectura SDN, sin embargo, no es un estándar establecido. La RFC 7426 define los planos presentes en una arquitectura SDN (figura 2.8):

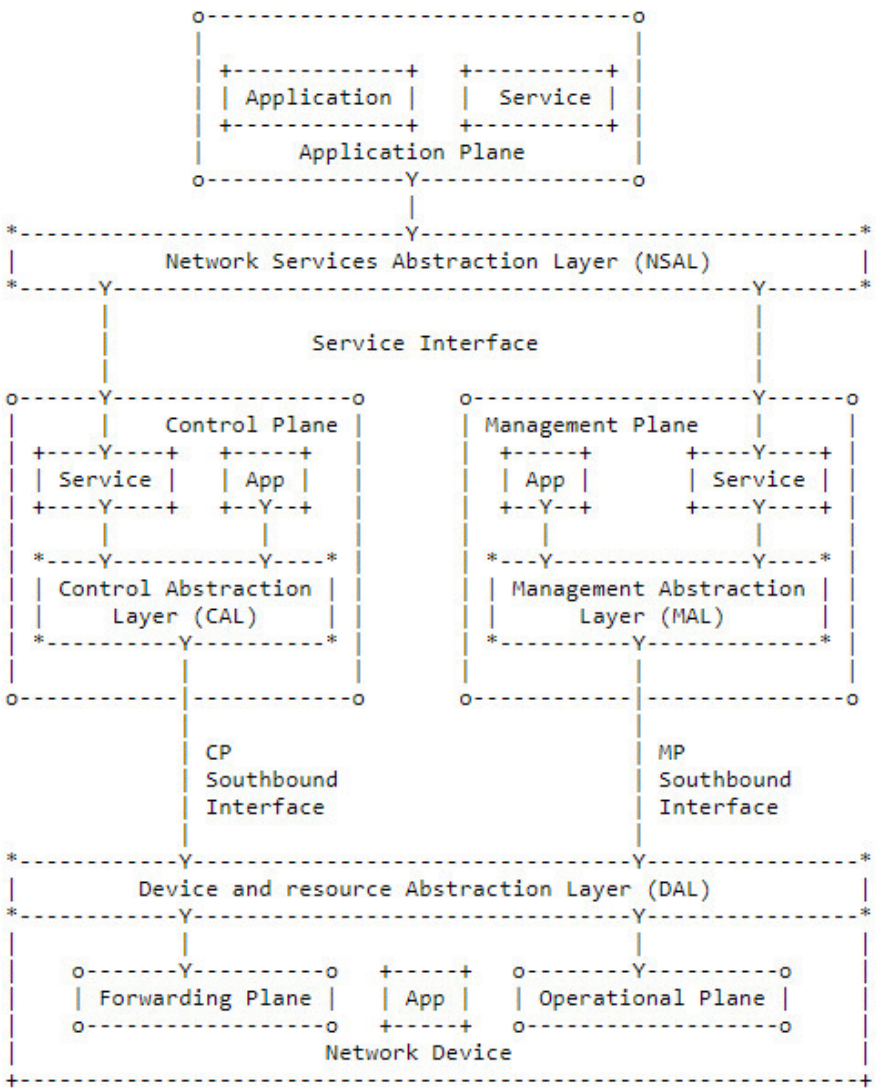

Figura 2.8: Arquitectura de la red SDN según la RFC 7426.
Fuente: HALEPLIDIS ed al. (2015)

- **Plano de reenvío (Forwarding Plane):** responsable de reenviar los paquetes de datos entrantes basado en las instrucciones recibidas desde el plano de control. Las acciones del plano de reenvío incluyen, pero no se limitan a, reenvío, descarte y medicación de paquetes.
- **Plano operativo (Operational Plane):** responsable de la gestión del estado operativo del dispositivo de red, por ejemplo, si el dispositivo está activo o inactivo, el número de puertos disponibles, el estado de cada puerto, etc. Según la RFC 7426, algunos participantes de la Internet Research Task Force (IRTF) opinan que el plano operacional no constituye un "plano" en sí, sino que forma parte del plano de reenvío.
- **Plano de control (Control Plane):** responsable de tomar decisiones sobre cómo los paquetes deben ser reenviados y de enviar dichas decisiones a todos los dispositivos de la red.
- **Plan de gestión (Management Plane**): responsable de supervisar, configurar y mantener los dispositivos de la red. Se enfoca principalmente en la gestión del plano operativo.

- **Plano de aplicación (Application Plane):** Es el plano donde se encuentran las aplicaciones y servicios que definen el comportamiento de la red.

La mayoría de los protocolos implementados para SDN (por ejemplo, Openflow de la Open Netwoking Foundation) no siguen la arquitectura propuesta en la RFC 7426. Otro esquema de arquitectura es el brindado por la **ITU-T (2014)** en el cual describe la arquitectura SDN en tres capas, de igual manera existen otros esquemas de arquitectura SDN de tres capas que brindan la misma idea general.

Según la **ITU-T (2014)**, la figura 2.9 explica de forma más genérica el enfoque SDN, el cual está constituido por tres planos: el plano de datos, el plano de control y el plano de aplicaciones. El plano de datos consiste en switches físicos y/o switches virtuales, en ambos casos los switches son responsables del reenvío de paquetes. La implementación interna del switch, como por ejemplo el buffer, parámetros de prioridad y otras estructuras de datos relacionadas con el reenvío pueden depender del proveedor o fabricante. Sin embargo, cada switch debe implementar un modelo de abstracción de reenvío de paquetes que sea común y abierto a los controladores SDN, este modelo se define en términos de una interfaz de programación de aplicaciones (API) abierta entre el plano de control y el plano de datos llamado Southbound API, el ejemplo más destacado es el protocolo OpenFlow. Una API de código abierto o estandarizado puede asegurar la interoperabilidad del código de la aplicación y la independencia del proveedor o fabricante de los dispositivos.

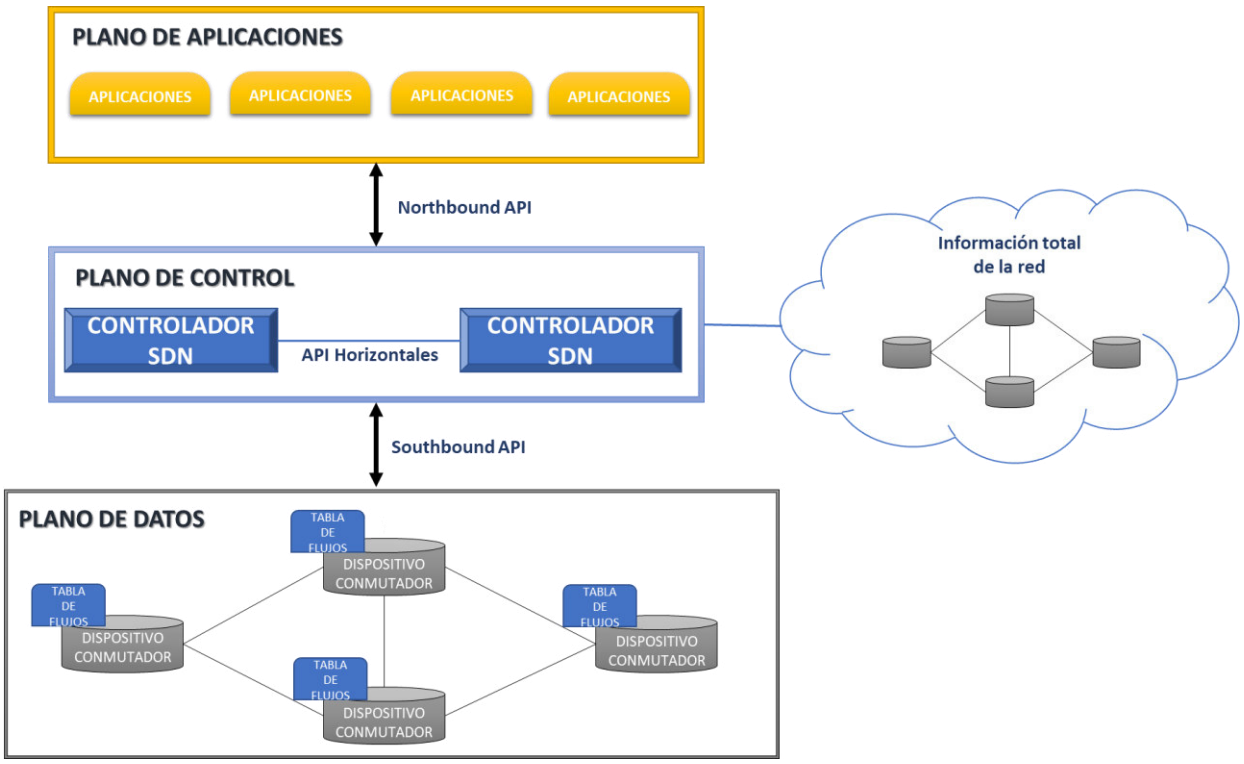

Figura 2.9: Arquitectura de la red definida por software basado en la recomendación.
Fuente: ITU-T (2014)

Según la **ITU-T (2014)**, el plano de control es en esencia el controlador SDN, estos controladores pueden implementarse directamente en un servidor físico o virtual. El controlador gestiona a los dispositivos en el plano de datos a través de las APIs como Openflow, además, los controladores tienen la capacidad de usar la información obtenida del dispositivo, como la capacidad de la red o la demanda de un servicio en especial con el fin de mejorar el flujo de datos de la red. Los controladores SDN también tiene un API que le permite interactuar con el plano de aplicaciones, son las llamadas Northbound APIs, el cual permite a los desarrolladores y administradores de redes implementar una amplia gama de aplicaciones de red personalizadas, pero aún no existe un Northbound API estandarizado ni un consenso sobre un Northbound API de código abierto.

La mayoría de los fabricantes ofrecen una API basado en "REpresentational State Transfer" (REST) para proporcionar una interfaz amigable a sus controladores SDN. También se tiene APIs Westbound/Eastbound, las cuales aún no están estandarizadas. Las APIs Westbound/Eastbound permiten la comunicación entre controladores SDN para sincronizar y tener un ambiente de alta disponibilidad de controladores.

Según **GORANSSON (2016),** en el plano de aplicación existen aplicaciones que interactúan con los controladores SDN. Las aplicaciones SDN son programas que utilizan la información obtenida por el plano de control del controlador para su toma de decisiones. Estas aplicaciones transmiten sus decisiones al controlador SDN a través de una Northbound APIs.

2.6.1. INTERFACES SDN

2.6.1.1. INTERFAZ SOUTHBOUND API

STALLINGS (2015) indica que las interfaces Southbound API se utilizan para la comunicación entre el controlador SDN y los dispositivos de la red (Switch, Router, etc.), pudiendo ser de código abierto o propietarios.

Según **STALLINGS (2015)**, las Southbound API permite tener el control de los dispositivos de la red ya que proveen de información al controlador SDN. Las "Southbound APIs" definen la forma en que el controlador SDN interactúa con el plano de datos de los dispositivos de la red, ajustando parámetros para responder dinámicamente de acuerdo con las demandas y/o necesidades de la red en tiempo real. El protocolo OpenFlow, que fue desarrollado por la Fundación Open Networking (ONF), es la interfaz Southbound más comúnmente aceptada en la comunidad SDN. Además de OpenFlow, existen otros protocolos que buscan el mismo fin, como el protocolo propietario OpFlex desarrollada por Cisco, el protocolo de configuración de red (NetConf) que utiliza el lenguaje XML para comunicarse con los switches y routers, el protocolo LISP (RFC 6830) también usado para proporcionar comunicación entre el controlador y los dispositivos de red.

2.6.1.2. INTERFAZ NORTHBOUND API

STALLINGS (2015) indica que las interfaces Northbound API se utilizan para la comunicación entre el controlador SDN y los servicios/aplicaciones que se ejecutan a través de la red.

Según **STALLINGS (2015)**, las Northbound APIs deben soportar una amplia gama de aplicaciones, ya que deben de comunicar eficientemente las necesidades de diferentes aplicaciones a través de la programación de la red SDN. Es por este motivo que no se ajusten a todas las aplicaciones y que aún no se tenga un protocolo estandarizado, actualmente existen varios protocolos para controlar diferentes tipos de aplicaciones a través de un controlador SDN. Las Northbound también se utilizan para integrar el controlador SDN con otras plataformas de virtualización, por ejemplo, OpenStack, vCloudDirector de VMware o CloudStack de código abierto.

2.7.PROTOCOLO OPENFLOW

La Open Networking Foundation (ONF) estandarizo el protocolo Openflow, el cual está definido en los documentos Openflow Switch Specification de la ONF.

Openflow es hoy el único protocolo no propietario, de propósito general, responsable de la comunicación del SouthBound API. Define la comunicación entre un controlador Openflow y un switch Openflow a través de mensajes. Permite al controlador SDN crear, modificar o eliminar las tablas de flujos (Flow-Tables) de los switches Openflow (físicos o virtuales) de forma dinámica y lograr implementar servicios como Firewalls, NAT, QoS, etc. Cuando el controlador define un flujo, proporciona al switch la información necesaria para saber cómo encaminara los paquetes entrantes que coincidan con ese flujo.

Las especificaciones el protocolo Openflow ha estado en constante cambio desde la primera liberación Openflow 0.2.0 el 28 de marzo de 2008, hasta la versión 1.5.1 liberado en 2015.

2.7.1. TABLAS DE FLUJO

Según las especificaciones del protocolo Openflow versión 1.3.5 de la **ONF (2015)**, una tabla de flujo consiste en varias entradas de flujo, las cuales son las mencionadas en la tabla 2.1:

Match Fields (Campos coincidentes)	Priority (Prioridad)	Counters (Contadores)	Instructions (Instrucciones)	Timeouts	Cookie

Tabla 2.1: Principales componentes de una entrada de flujo.
Fuente: ONF (2015)

Según la **ONF (2015)**, una entrada de flujo consiste en:

- **Campos coincidentes (Matchs Fields):** Se utilizan como criterios de coincidencia para determinar si un paquete entrante coincide con esta entrada. Si existe una coincidencia, entonces el paquete pertenece a este flujo.
- **Prioridad (Priority):** Nivel de prioridad que tendrá la entrada frente a otras entradas que también coincidan con un paquete determinado.
- **Contadores (Counters):** Los contadores se utilizan para realizar un seguimiento de las estadísticas relativas a este flujo, por ejemplo, cuántos paquetes han sido reenviados o eliminados para este flujo.
- **Instrucciones:** Los campos de instrucciones indican qué debe hacer el switch con un paquete que coincida con la entrada coincidente.
- **Timeouts:** Máximo tiempo que la entrada estará en la tabla de flujos, si el valor es 0 nunca expirara.
- **Cookie:** Dato que envía el controlador por motivos estadísticos, no se usa para el procesado de paquetes.

Cuando un paquete llega al switch OpenFlow desde un puerto de entrada (o, en algunos casos, desde el controlador), el paquete es comparado con la tabla de flujo para determinar si hay una entrada de flujo coincidente.

Según la **ONF (2015)**, los siguientes campos de coincidencia asociados con el paquete entrante se pueden utilizar para coincidir con la entrada de flujo:

- Puerto de entrada
- VLAN ID
- Prioridad de la VLAN
- Dirección Ethernet de origen.
- Dirección Ethernet de destino.
- Tipo de trama ethernet.
- Dirección IP de origen.
- Dirección IP de destino.
- Tipo de protocolo IP (IPv4 o IPv6)
- IP Type of Service (ToS) bits.
- TCP/UDP puerto de origen.
- TCP/UDP puerto de destino.

2.7.2. PROCESO DE COINCIDENCIA (MATCH)

El switch Openflow realiza el procedimiento descrito en la figura 2.10 cuando recibe un paquete de datos.

Según la **ONF (2015),** el switch Openflow realiza el procedimiento descrito en la figura 2.10 cuando recibe un paquete de datos. El switch empieza a buscar coincidencia en la primera tabla de flujo según los criterios configurados (generalmente la tabla de flujo N° 0), el switch comparara el paquete con las entradas de flujo de la tabla de flujo y solo seleccionara la entrada de flujo de mayor prioridad que coincida con el paquete.

Cuando se encuentra la coincidencia, los contadores asociados con la entrada de flujo se actualizarán y el switch ejecutara la acción definida en la entrada de flujo coincidente. Si

hubiera múltiples entradas de flujo coincidentes con la misma prioridad más alta, la entrada del flujo es categorizada como explícitamente indefinida y no serán tomadas en cuenta.

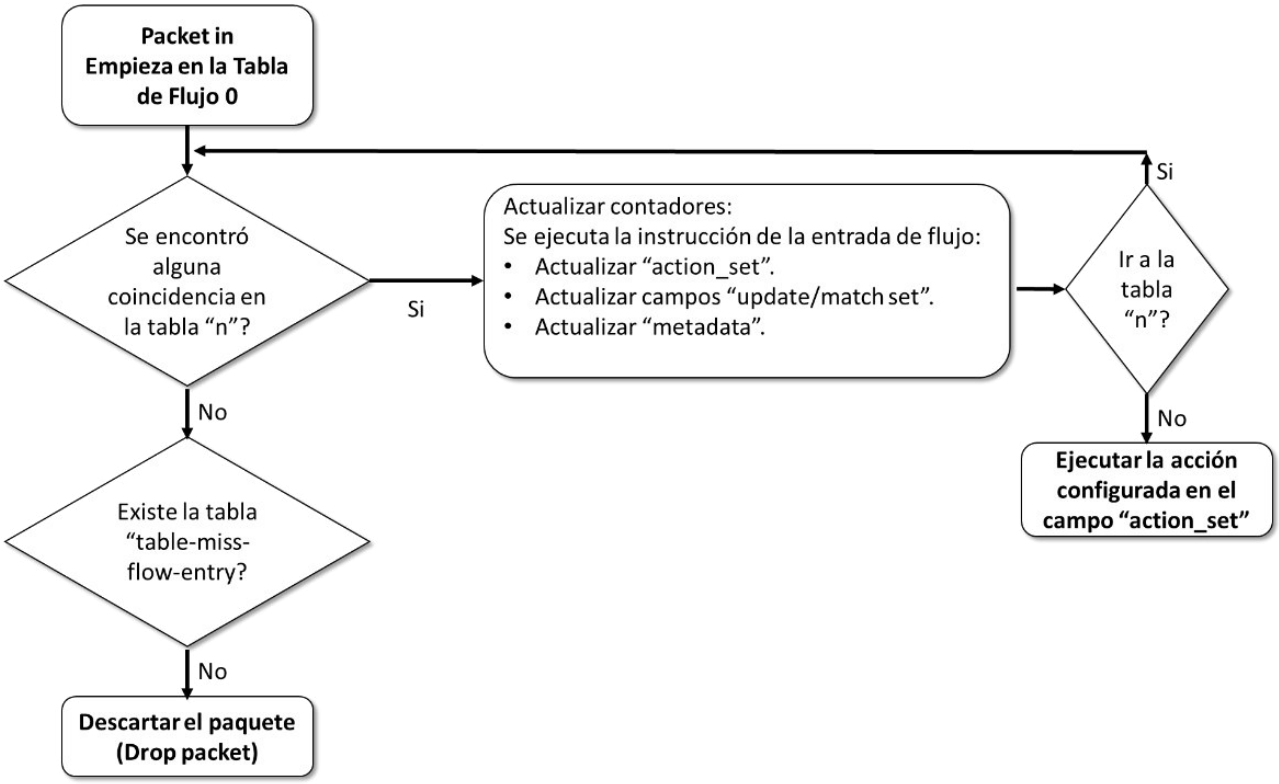

Figura 2.10: Diagrama de flujo del proceso de coincidencia en un switch Openflow.
Fuente: ONF (2015)

2.7.3. ENTRADA DE FLUJO "TABLE-MISS"

Según la **ONF (2015),** los switches Openflow deben soportar una entrada de flujo llamada "Tabla-miss". Esta entrada de flujo especifica cómo procesar paquetes que no coinciden con otras entradas de flujo en la tabla de flujo, y puede, por ejemplo, enviar paquetes al controlador o descartar el paquete.

La entrada de flujo "Tabla-miss Flow" se identifica por tener activa todos los campos de coincidencia y tiene la prioridad más baja (prioridad igual a cero). Esta entrada se comporta de la misma manera que cualquier otra entrada de flujo, por lo cual, no existe de manera predeterminada en una tabla de flujo, el controlador puede agregarla o eliminarla en cualquier momento. Si la entrada de flujo "Tabla-miss" no existe, los paquetes predeterminados que no coinciden con las entradas de flujo se descartan.

2.7.4. ELIMINACIÓN DE LAS ENTRADAS DE FLUJO

Según la **ONF (2015),** las entradas de flujo se eliminan de las tablas de flujo de dos maneras, por órdenes del controlador o a través del mecanismo de expiración de la entrada de flujo.

- Por mecanismos de expiración:

Según la **ONF (2015),** el mecanismo de expiración del flujo es controlado por el switch independientemente del controlador y depende de los tiempos configurados en cada entrada de flujo. Una entrada de flujo tiene configurado dos mecanismos de expiración, el idle_timeout y el hard_timeout

El campo hard_timeout define el tiempo que estará instalado una entrada de flujo en el switch de manera explícita. Si la hard_timeout es distinto de cero, el switch eliminara la entrada de flujo después que supere el número de segundos configurado en el hard_timeout aun cuando la entrada de flujo tenga paquetes coincidentes. Si el campo hard_timeout es cero, no será considerado.

El campo idle_timeout define el tiempo en que el switch eliminara la entrada de flujo si es que no tiene paquetes coincidentes en el tiempo especificado. Si la idle_timeout es distinto de cero, la entrada de flujo será eliminada cuando no tenga coincidencia de paquetes en el número de segundos especificado. Si el campo idle_timeout es cero, no será considerado.

Una entrada de flujo no será removida si el idle_timeout y el hard_timeout son iguales a cero.

- Por órdenes del controlador:

Según la **ONF (2015),** el controlador puede eliminar activamente las entradas de flujo de las tablas de flujo mediante el envío de mensajes de modificación de la tabla de flujo.

2.7.5. COMPONENTES DEL SWITCH OPENFLOW

Según las especificaciones del protocolo brindadas por la **ONF (2015)**, un Switch Openflow se compone de tres características fundamentales (Figura 2.11):

- **Tablas de flujos:** Indican como debe ser procesado un flujo determinado de paquetes.

- **Canal seguro de comunicación hacia el controlador:** El canal de comunicación entre el switch y el controlador es el protocolo Openflow, este debe ser muy seguro ya que el controlador gestiona toda la red de forma centralizada.

- **Controlador SDN:** El Controlador SDN es el encargado de gestionar toda la red de forma centralizada, por tanto, debe actualizar las tablas de flujos ante la llegada de nuevos paquetes, además que permite crear, añadir o eliminar las entradas de flujos.

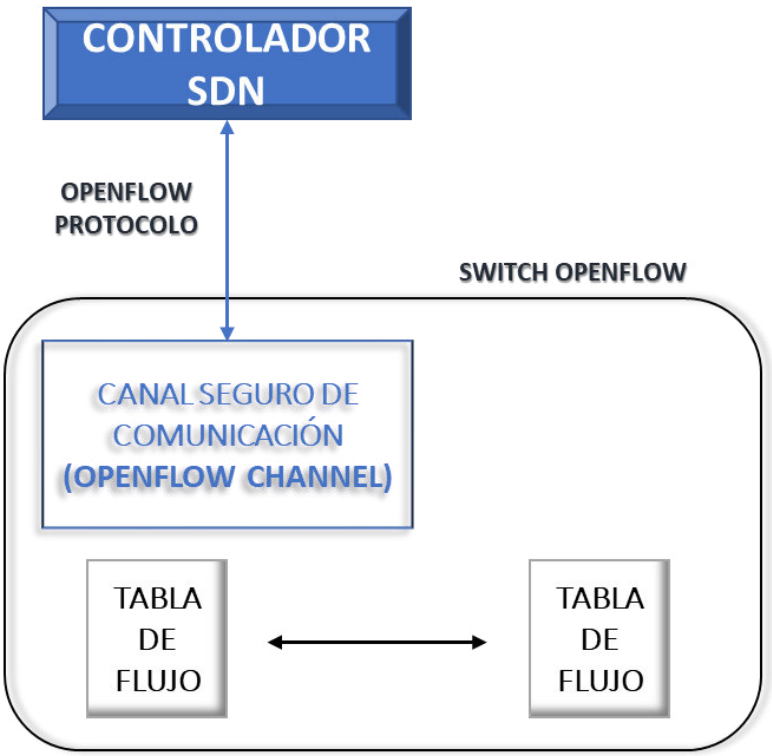

Figura 2.11: Principales componentes de un Switch Openflow.
Fuente: ONF (2015)

Cabe decir que un Switch Openflow puede ser cualquier dispositivo que pueda ser controlado a través de controladores SDN. Un switch OpenFlow puede ser OpenFlow-only o OpenFlow-hybrid. Un switch OpenFlow-Only reenvía paquetes utilizando solamente las tablas de flujo, en cambio, un switch OpenFlow-hybrid es un dispositivo que también puede conmutar paquetes utilizando la lógica de un Router o switch tradicional, usando su propio plano de control. Un switch OpenFlow-hybrid requiere un mecanismo de clasificación que determine si el paquete será procesado mediante los mecanismos de OpenFlow o si usara el procesamiento tradicional de paquetes.

2.7.6. PROCESAMIENTO DEL SWITCH OPENFLOW

Como cualquier switch, la función principal de un switch Openflow es la de conmutar paquetes, tomando el paquete de entrada que ingresa en un puerto y reenviarlo a través de otro puerto, realizando las modificaciones que se tengan que realizar al paquete antes de reenviarlo.

Según la **ONF (2015)**, el switch Openflow debe ser capaz de ejecutar las siguientes opciones fundamentales al paquete de entrada:

- Reenvío de paquetes a uno o varios puertos determinados: La acción consiste en encaminar el paquete a otros puertos a velocidad de línea.

- Encapsulación y reenvío del paquete al controlador: Se usa cuando el switch openflow recibe un paquete que no coincide con ninguna entrada de flujo. El paquete se reenviará por un canal seguro hacia el controlador para que determine la accione que se debe realizar al paquete.

- Descartar el paquete: El switch OpenFlow debe ser capaz de bloquear paquetes sospechosos, o descartarlos si eso indica la tabla de flujo.

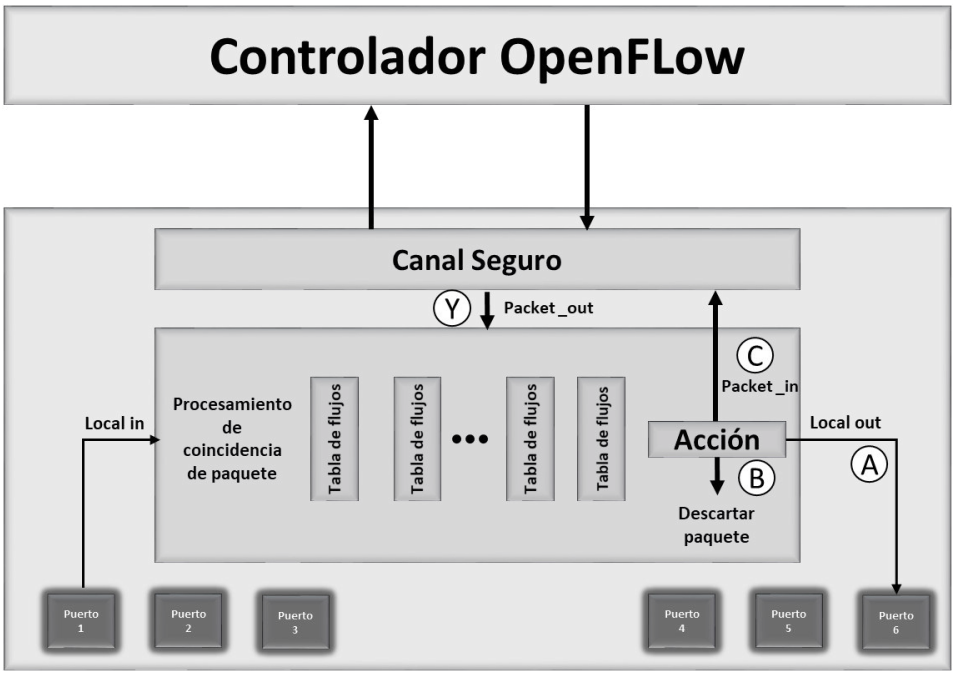

Figura 2.12: Procesamiento del switch Openflow.
Fuente: GORANSSON (2016)

Según **GORANSSON (2016),** el procesamiento que utiliza el switch para reenviar el paquete se observa en la figura 2.12. Por ejemplo, el paquete que ingrese al switch por alguno de sus puertos será enviado a la primera tabla de flujo. Si el switch encuentra alguna coincidencia en una entrada, se realizará la acción descrita en el campo "Instrucciones". Seguidamente el paquete irá pasando por las tablas de flujos restantes y modificará constantemente el valor de "Action-Set". Finalmente, el switch ejecutará el contenido definido en "Action-Set" de forma ordenada (figura 2.13) y decidirá a que puerto enviara el paquete (ruta “A”) o si lo descartara (ruta “B”).

Si un paquete no encuentra ninguna coincidencia en las tablas flujo deberá ser enviado al controlador SDN siguiendo la ruta “C”. El controlador definirá un nuevo flujo para el paquete y enviará al switch una o más entradas para las tablas de flujo existente utilizando el mensaje Packet_Out (ruta “Y”). De esta manera, en la siguiente ocasión que el switch reciba un paquete similar se tendrá una coincidencia con el nuevo flujo y el switch no tendrá que volver a reenviar el paquete hacia el controlador, sino que serán encaminados de acuerdo con la entrada de flujo coincidente.

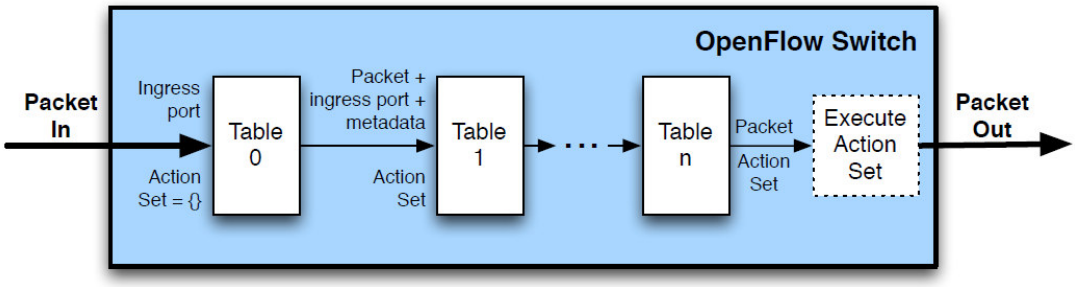

Figura 2.13 Seguimiento del paquete a través de las tablas de flujo.
Fuente: ONF (2015).

Según la **ONF (2015)**, las tablas de un switch Openflow están numeradas secuencialmente, comenzando en 0. El procesamiento de la canalización siempre comienza en la primera tabla: el paquete se compara primero con las entradas de flujo de la tabla de flujos 0. Pueden usarse otras tablas en función del resultado de la coincidencia de la primera tabla.

2.7.7. MENSAJES DEL PROTOCOLO OPENFLOW

Según la **ONF (2015)**, los mensajes de Openflow v1.3.5 se pueden clasificarse en tres tipos:

- Mensajes del controlador SDN al switch.
- Mensajes asíncronos.
- Mensajes simétricos.

2.7.7.1. MENSAJES DEL CONTROLADOR SDN AL SWITCH

Son los mensajes enviados e inicializados por el controlador SDN hacia el switch Openflow para gestionar y/o conocer el estado del switch. La **ONF (2015)** nos detalla los siguientes mensajes:

- **Features:** Este mensaje le permite al controlador SDN solicitar la identidad y las capacidades básicas del switch. El switch al recibir este mensaje debe responder especificando sus características.

- **Configuration:** Este mensaje le permite al controlador SDN enviar parámetros de configuración en el switch.

- **Modify-State:** Este mensaje le permite al controlador SDN puede gestionar las tablas de flujos en el switch, permitiendo añadir, eliminar o modificar flujos de entrada.

- **Read-State**: Este mensaje le permite al controlador SDN para recoger información estadística, tales como la configuración actual.

- **Packet-out:** Este mensaje le permite al controlador SDN designar el puerto del switch por donde se encaminará el paquete, normalmente este mensaje es enviado como respuesta a un "Packet-in" enviado por el switch. El mensaje debe contener también una lista de acciones que se aplicarán en el orden en que se especifiquen; si no se especifica ninguna acción significa que el paquete será descartado.

- **Barrier:** Mensaje enviado por el controlador SDN para informar que determinadas operaciones se han completado.

- **Role-Request:** Este mensaje es usado principalmente si un switch se conecta a varios controladores SDN en un esquema de alta redundancia. Usa para fijar el rol del controlador SDN.

2.7.7.2. MENSAJES ASÍNCRONOS

Son los mensajes enviados e inicializados por el switch Openflow para informar al controlador SDN sobre eventos en la red y cambios en el estado del switch. La **ONF (2015)** nos detalla los siguientes mensajes:

- **Packet-in:** Este mensaje es usado por el switch para transferir el control del paquete recibido al controlador SDN, mayormente porque no encuentra un flujo asociado al paquete, pero también puede ser debido a una acción expresa de un flujo de entrada. La respuesta del controlador SDN ante este mensaje es un paquete "Packet-out".

- **Flow-Removed:** Este mensaje informa al controlador SDN de la eliminación de una entrada en una tabla de flujo.

- **Port-Status:** Este mensaje informa al controlador SDN de un cambio en un puerto, posiblemente por una nueva configuración o porque el puerto paso a un estado inactivo (down).

2.7.7.3. MENSAJES SIMÉTRICOS

Son los mensajes que pueden ser inicializados por switch Openflow o por el controlador SDN sin necesidad de autorización previa. La **ONF (2015)** nos detalla los siguientes mensajes:

- **Hello:** Los mensajes "Hello" son intercambiados entre el Controlador SDN y el siwtch al principio de una conexión OpenFlow.

- **Echo:** Un mensaje "Echo request" puede ser enviado por el controlador SDN o el switch, dispositivos y el otro extremo de la conexión deberá enviar un "Echo reply" para validar que el mensaje fue recibido. Se usan comúnmente para validar la conexión entre ambos dispositivos.

- **Error:** Son mensajes de error notificados por el controlador SDN o por el switch cuando existen problemas de conexión. Es comúnmente usado por el switch para notificar un fallo de una petición iniciada por el Controlador.

- **Experimenter:** Mensajes de experimentación se usan para proveer funcionalidades adicionales de forma estándar o para futuras características del protocolo Openflow.

2.7.8. ESTABLECIMIENTO DE LAS SESIONES

Según **ONF (2015)**, los mensajes entre el controlador y el switch anteriormente descritos se envían a través de un canal seguro previamente establecido. El canal seguro es una conexión TCP en donde se utilizada TLS (Transport Layer Security) para la comunicación segura. Los mensajes empiezan por el encabezado Openflow, en el encabezado se especifica el número de versión de Openflow, el tipo de mensaje, la longitud del mensaje y el ID de del mensaje (figura 2.14).

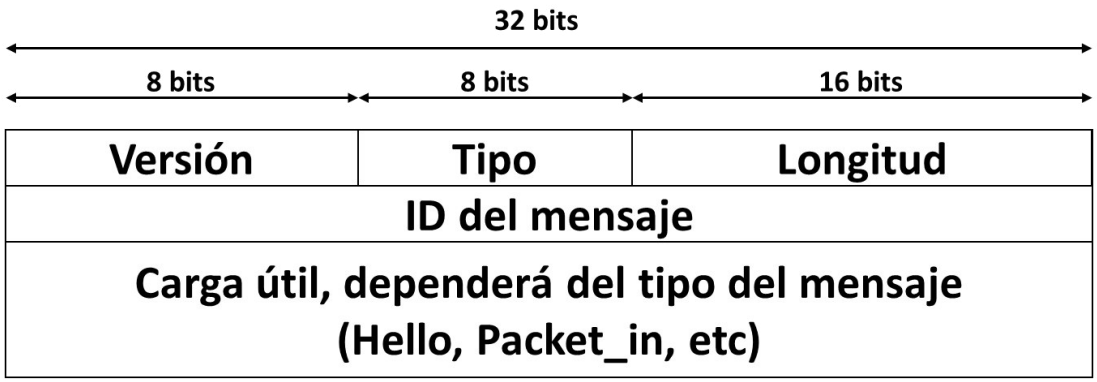

Figura 2.14: Estructura de un mensaje Openflow
Fuente: **ONF (2015)**.

Según **GORANSSON (2016),** en la figura 2.15 se observa los mensajes generalmente enviados durante las fases de inicialización, operación o monitoreo de la comunicación entre el switch y el controlador. Las fases de operación y monitoreo ocurren de forma simultánea en la mayoría de los casos. Después del establecimiento de la sesión del túnel TLS, se intercambian mensajes Hello para determinar el número de versión de OpenFlow más alto soportado entre el switch y el controlador, luego se intercambian los mensajes Features para que el controlador conozca las características del switch. El mensaje unidireccional, Set_Config, es enviado por el controlador para establecer parámetros de configuración en el conmutador y así crear, modificar o eliminar las entradas de flujo existentes en el switch a través del mensaje FLOW_MOD.

Durante la etapa de monitoreo, los mensajes ECHO son utilizados tanto por el switch como por el controlador para comprobar que la sesión entre ambos aún sigue establecida, además de medir la latencia actual o el ancho de banda de la conexión. Otros mensajes de estado es el Flow_Removed usado para informar que la tabla de flujo ha sido eliminada en el switch y el Port_Status para indicar el estado del puerto, sea por un cambio físico en el propio medio de comunicación o intervención del usuario.

Durante la etapa de operación, los mensajes comunes son los PACKET_OUT y PACKET_IN mencionado anteriormente.

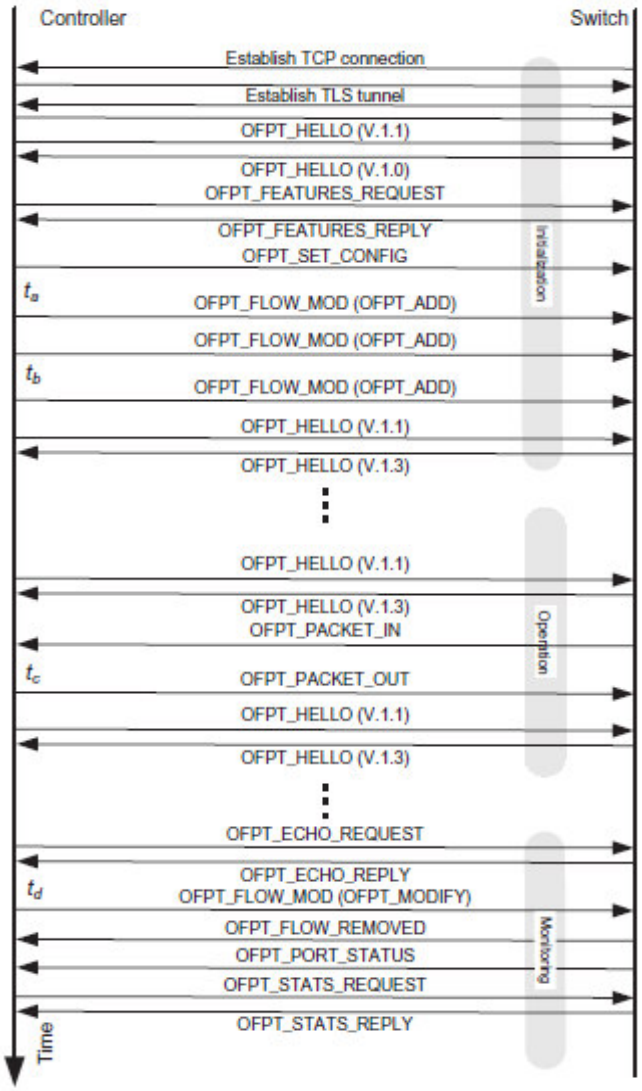

Figura 2.15: Establecimiento de las sesiones Openflow.
Fuente: GORANSSON (2016)

2.7.9. CANAL SEGURO OPENFLOW

Según **ONF (2015)**, el canal seguro se refiere al medio físico por el cual se envía el tráfico de control, puede estar conformador por una red exclusiva para este fin o usar la misma infraestructura de la red de tráfico de datos. Ambas formas son las siguientes:

- Out-Of-Band Control.
- In-Band Control.

2.7.9.1. OUT-OF-BAND CONTROL

Según **ONF (2015)**, el control fuera de banda, o Out-of-band control, utiliza puertos Ethernet y enlaces separados (es decir, una red independiente) para conectar los switches al controlador. La infraestructura separada puede ser física o lógica. En el primer caso, los switches Openflow tienen un puerto de "gestión" físico que debe estar conectado al controlador o a través de una red tradicional conectada al controlador. En el segundo caso, se utiliza túneles o redes lógicas independiente en la misma infraestructura física, por ejemplo, en el despliegue original de Openflow en Stanford, los switches se configuraron con 3 VLANs, uno para el control de tráfico Openflow, uno para experimentar tráfico de datos con Openflow, y uno para el tráfico de producción.

2.7.9.2. IN-BAND CONTROL

Según **ONF (2015)**, el control en banda, o In-band control, utiliza los mismos enlaces (es decir, la misma red) tanto para el tráfico de datos como para el tráfico de control. Este caso generalmente requiere que los conmutadores tengan un conjunto predefinido de reglas para establecer la conexión con el controlador.

2.8.SDN PLANO DE DATOS

El plano de datos, también llamado como plano de usuario, plano de reenvío, plano portador o como plano de reenvío (Forwarding Plane). Según la **UIT (2014**), es la parte de una red que transporta tráfico de usuarios y está conformado por los dispositivos de reenvió. La toma de decisiones y el procesamiento de los paquetes se llevan a cabo de acuerdo con lo establecido por el plano de control SDN.

Un dispositivo de red en una red SDN realiza la función simple de reenvío de acuerdo con las tablas de flujo establecidas, sin software incorporado para tomar decisiones autónomas. Otras funciones es la de modificar el encabezado de los paquetes de acuerdo con las necesidades de reenvió, así como descartar el paquete. Como ya se ha explicado, el Southbound API que más aceptado es el protocolo Openflow utilizado para la comunicación entre el plano de control y el plano de datos.

2.9.SDN PLANO DE CONTROL

Según la **ITU (2014),** El plano de control es el encargado de administrar y tener la visión de la totalidad de la red, usa esta información para proporcionarla a una aplicación, permitiendo a la aplicación realizar cambios en la red de forma dinámica de acuerdo con el comportamiento de la red.

El plano de control es principalmente el controlador SDN centraliza que, además de tener la visión total de la red, implementa decisiones de políticas, controla todos los dispositivos SDN mediante la Southbound API y además tiene la Northbound API encargada de la comunicación con el plano de aplicaciones.

Según la **ITU (2014),** el plano de aplicación puede dividirse en las siguientes sub-planos:

- Soporte de aplicaciones

  Brinda la función del soporte de aplicaciones y proporciona una interfaz de control de aplicaciones SDN (Northbound API). Mediante esta interface de control entre el plano de datos y la aplicación, permite a las aplicaciones acceder a información de la red que necesitan.

- Coordinación:

  Proporciona el control automatizado, la gestión de recursos de red y la coordinación entre las solicitudes de la capa de aplicación y los recursos de red que necesiten.

- Abstracción

  La función de abstracción interactúa con los recursos de la red y proporciona una abstracción de los recursos de red, incluidas las capacidades y características de la red, con el fin de admitir la gestión y coordinación de recursos de red físicos y virtuales.

2.9.1. CONTROLADOR SDN

Según la **ITU (2014),** el controlador SDN es responsable de la abstracción la red, controlar los dispositivos SDN y brindar la información de los recursos de red a las aplicaciones SDN. El controlador permite que la aplicación SDN defina flujos en dispositivos y ayude a la aplicación a responder a paquetes que son enviados al controlador por los dispositivos SDN. El controlador suele ser una máquina de alto rendimiento ya que tiene que controlar y administrar un gran número de dispositivos de red.

Según **GORANSSON (2016)** la arquitectura del controlador está constituida por módulos y APIs de comunicación (figura 2.16). Los modelos brindan las funcionalidades propias del controlador, los Northbound API que pueden estar basados en REST, Phyton, Java, etc., y el protocolo Openflow es el Southbound API más conocido.

2.9.1.1. MÓDULOS

Los módulos del controlador representan sus principales funciones, las cuales son el descubrimiento y seguimiento de dispositivos y topologías, la gestión de flujos, la gestión de dispositivos y el seguimiento de estadísticas. Para la realización de sus funciones, los módulos mantienen una base de datos locales para contener la topología y estadísticas de la red.

Según **GORANSSON (2016)**, las principales funciones del controlador son:

- Descubrimiento de los dispositivos de usuarios finales: Tales como ordenadores portátiles, escritorios, impresoras, dispositivos móviles, etc.

- Detección de dispositivos de red: Descubrimiento de dispositivos de red que comprenden la infraestructura de la red, tales como switches, routers, etc.
- Administración de los dispositivos de la red: Mantiene la información sobre los detalles de interconexión entre los dispositivos de red y sobre los dispositivos de usuario final a los que están conectados directamente.
- Gestión de flujo: Mantiene una base de datos de los flujos gestionados por el controlador y realiza las coordinaciones necesarias con los dispositivos de red para garantizar la sincronización de las entradas de flujo en el dispositivo de red con la base de datos del controlador.

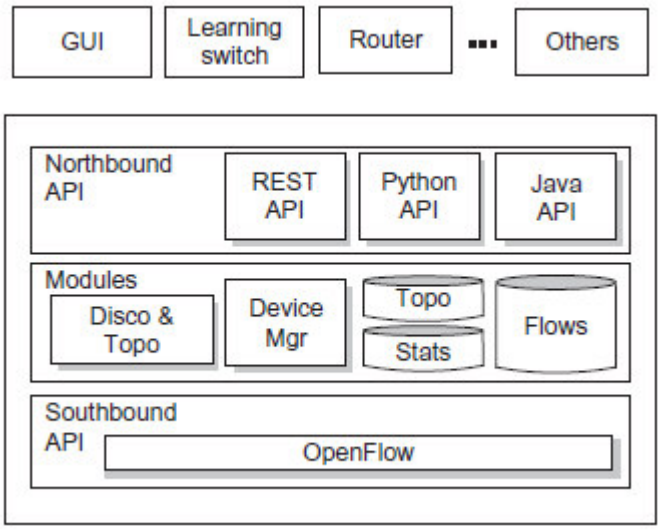

Figura 2.16: Arquitectura de un controlador SDN.
Fuente: GORANSSON (2016)

2.9.2. CONTROLADORES SDN DE CÓDIGO ABIERTO

Actualmente se cuenta con controladores tanto desarrollados por proveedores como Cisco, VMware, etc., y de código abierto. **GORANSSON (2016)**, describe algunos controladores de código abierto:

- **OpenDaylight (ODL):** Fue desarrollada por Cisco e IBM, y está escrito en Java. OpenDaylight es capaz de desplegarse en una variedad de entornos de red de producción, puede proporcionar soporte para otros estándares SDN y protocolos futuros.

- **Open Nework Operating System (ONOS):** Un SDN NOS de código abierto, lanzado inicialmente en 2014 y es proyecto sin fines de lucro apoyado por los proveedores de servicios Nivel 1 -AT & T, NTT Communications, SK Telecom, los principales proveedores Ciena, Cisco, Ericsson, entre otros.

- **POX**: Es una plataforma de desarrollo de código abierto para aplicaciones de control de redes SDN basadas en Python. POX tiene una API y documentación

bien escrita, también proporciona una interfaz gráfica de usuario basada en la web (GUI).

- **Floodlight:** Controlador de código abierto desarrollado por Big Switch Networks, basado en Apache Ant. Floodlight proporciona una interfaz GUI basada en web y basada en Java y la mayor parte de su funcionalidad está expuesta a través de una API REST.

- **Ryu:** Un framework basado en SDN basado en componentes de código abierto desarrollado por NTT Labs. Es de código abierto y totalmente desarrollado en python.

Controlador	Fuente	Licencia	Lenguaje
OpenDayLight	OpenDayLight	EPL	Java
ONOS	ON.Lab	Apache 2.0	Java
NOX	ICSI	GPL	C++
POX	ICSI	GPL	Python
Beacon	Stanford University	GPLv2, Stanford FOSS	Java
Floodlight	Big Switch Networks	Apache 2.0	Java
Ryu	NTT Communications	Apache 2.0	Python
FLowER	Travelping GmbH	MIT License	Erlang
Jaxon	University of Tsukuba	GPLv3	Java
Mul SDN	Kulcloud	GPLv2	C
NodeFlow	Gary Berger	-	Javascript
Trema	NEC	GPLv2	Ruby.C

Tabla 2.2: Principales controladores SDN de código abierto.
Fuente: Elaboración propia, basado en GORANSSON (2016)

2.10. SDN PLANO DE APLICACIONES

Según la **ITU (2014),** la capa de aplicación es donde las aplicaciones SDN especifican los servicios de red o aplicaciones empresariales ya que definen el comportamiento de los recursos de la red de manera dinámica y programable.

Las aplicaciones SDN especifican cómo deben controlarse y asignarse los recursos de red, interactuando con la capa de control SDN a través de interfaces de control de aplicaciones. La programación de una aplicación SDN hace uso de la vista abstracta de los recursos de red proporcionados por la capa de control SDN por medio de información y modelos de datos expuestos a través de la interfaz de control de aplicación.

2.10.1. APLICACIONES SDN

Las aplicaciones SDN pueden ser desarrolladas para cumplir cualquier función para la que se diseñó, utilizando la información obtenida por la capa de control (Controlador

SDN) puede realizar las actividades de un firewall, monitoreo de la red, un balanceador de carga, entre otras posibilidades.

Las aplicaciones SDN responden a eventos derivados del controlador SDN o de entradas externas. Como nos menciona **GORANSSON (2016)**, las aplicaciones responden a los siguientes eventos:

- **Descubrimiento de dispositivos de usuario final:** Los eventos son enviados a la aplicación SND cuando se descubre un nuevo dispositivo de usuario final, el cual puede ser una laptop, pc, smartphone, etc. En estos casos la aplicación decidirá qué hacer con la nueva dirección MAC aprendida.

- **Descubrimiento de dispositivos de red:** Los eventos son enviados a la aplicación SND cuando se conecta un nuevo dispositivo de red a la topología, pudiendo ser un switch, router, etc. La respuesta de la aplicación podría ser descargar un conjunto de entradas de flujo predeterminadas al dispositivo recién descubierto.

- **Paquete de entrada:** Los paquetes de entrada se envían a la aplicación SND cuando una entrada de flujo instruye al dispositivo SDN a reenviar el paquete al controlador o porque no hay una entrada de flujo coincidente en el dispositivo SDN o podría descartar el paquete.

CAPITULO III

ANÁLISIS DE LA GESTIÓN EN REDES TRADICIONALES Y TOPOLOGÍA DE LA RED TELEMÁTICA

3.1. RED TELEMÁTICA – RED ARQUITECTURA TRADICIONAL

El Campus de la Universidad Nacional Mayor de San Marcos está estructurado por varias facultades, bibliotecas y dependencias administrativas descentralizadas, siendo la Red Telemática la dependencia administración que tiene la función principal de la administración de la red LAN Campus y de los principales servicios de red. La red LAN de la Red Telemática consta de computadores, tarjetas de interfaz de red, dispositivos periféricos, dispositivos de red, entre otros, interconectados para brindar servicios a los usuarios finales.

La red LAN permite el acceso a los servicios internos a múltiples usuarios. Según la **Red Telemática (2017),** la Red Telemática utiliza los recursos de la red brindar los siguientes servicios de red:

- Hosting: La Red Telemática ofrece a las facultades, departamentos y grupos de Investigación, la posibilidad de alojar sus páginas Web en los servidores de esta institución. Igualmente se ofrece este servicio de alojamiento a las unidades administrativas, teniendo en cuenta limitaciones propias de un servicio de alojamiento compartido.
- Correo institucional: La UNMSM, en colaboración con Google, ofrece una versión especial de las aplicaciones de Google para la comunidad universitaria. Entre los servicios que se ofrecen están: Correo electrónico, Calendar y Drive entre otros.
- Telefonía: La UNMSM dispone de un sistema telefónico basado en IP (VoIP). El sistema cuenta con una amplia gama de servicios añadidos como buzones de voz, identificación de llamadas entrantes, llamadas perdidas, gestión de líneas ocupadas, conferencia múltiple, etc.
- Alojamiento de Servidores: El servicio de alojamiento de servidores consiste en proveer un espacio físico o virtual en el datacenter de la Red Telemática, cubriendo de esta manera las necesidades de las facultades y/o dependencias de la UNMSM.
- Acceso a internet y redes externas: La Red Telemática monitoreará las actividades de la red, tanto para correo electrónico, internet y uso de red de datos con el fin de vigilar el cumplimiento de las políticas establecidas para el uso de tecnologías de información.

La Red Telemática consta de un diseño de red modular, divide la red en áreas y módulos de red funcionales para mejorar la escalabilidad. Según **BRUNO, JORDAN (2017**), las áreas y módulos son:

- Área de la red campus.
- Módulo de data center
- Módulo de puerta de acceso WAN/Internet.
- Módulo de acceso remoto.

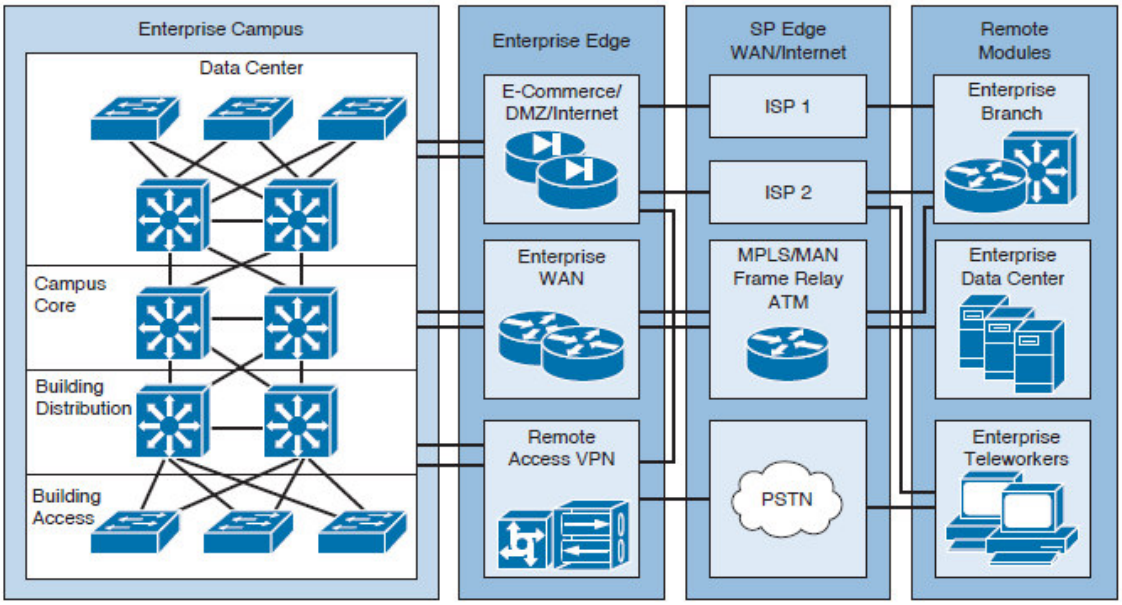

Figura 3.1: Red modular.
Fuente: BRUNO, JORDAN (2017)

3.2. CARACTERÍSTICAS TÉCNICAS MÍNIMAS DE LOS DISPOSITIVOS DE TELECOMUNICACIONES DEL CAMPUS DE LA UNMSM

Según los términos de referencia y requerimientos técnicos mínimos establecidos para el servicio de transporte de datos del campus de la UNMSM y de la red telemática en octubre del 2016, las características mínimas de los dispositivos de comunicaciones son los siguientes:

❖ Dispositivos Routers:

- Routing: BGP, OSPF, RIP v1/v2, Rutas estáticas, ECMP, RPF y enrutamiento basado en rutas y políticas.
- Multicast: IGMP v1/v2/V3, PIM-SM, PIM-DM, SSM y multicast dentro de un túnel IPSec.
- Alta disponibilidad: Activo/Activo, Activo/Pasivo, VRRP.
- Gestión de tráfico (QoS): Garantizar ancho de banda, máximo ancho de banda, políticas de ingreso de tráfico, priorización de utilización de ancho de banda, marcado DiffServ.

- Switching L2: LACP, VLAN 802.1Q y autenticación de puerto basada en 802.1x.
- Todos los equipos (routers) deberán tener habilitado una comunidad de lectura SNMP para que permita visualizar al personal técnico autorizado de la universidad, los eventos que ocurren en los equipos (routers). La recepción, visualización, y almacenamiento en servidor será responsabilidad de la UNMSM.
- La solución debe filtrar tráfico IP del tipo IPv4 e IPv6. Estas reglas deben ser configurables tanto por CLI (Command Line Interface, Interface de línea de comando) como por GUI (Graphical User Interface, Interface Gráfica de Usuario).
- La solución soporta ruteo estático, incluyendo pesos y/o distancias y/o prioridades de rutas estáticas.
- La solución soporta políticas de ruteo (policy based routing).
- La configuración de BGP debe soportar Autonomous System Path (AS-PATH) de 4 bytes.
- La solución de seguridad debe permitir la configuración de clústers en modo de operación en alta disponibilidad (HA), tanto para IPv4 como para IPv6.
- La solución debe ser capaz de operar en modalidad Alta Disponibilidad Activo-Pasivo y Activo-Activo
- La solución de seguridad debe ser capaz de definir al menos dos interfaces para sincronización.
- La solución de seguridad debe ser capaz de operar sin utilizar Multicast para el modo alta disponibilidad.
- La solución permite definir interfaces de gestión independientes para cada miembro en un clúster.

❖ Dispositivos switch de distribución LAN:

- Sistema operativo con actualización automática.
- Soporte para herramienta de análisis de flujo de tráfico (SFLOW, Netflow, o similar), integrada al Software de recolección de flujo de tráfico de la UNMSM.
- Procesos de debug para el análisis detallado de fallas.
- Protocolo VRRP
- Calidad de Servicio (QoS) 802.1p con DSCP.
- Protocolo Spanning Tree y mejoras tales como convergencia rápida (RST 802.1w y múltiples instancias (MST 802.1s).
- Manejo de VLANs por puerto y 802.1q (trunking).
- Tráfico Multicast IGMP v2 y v3 snooping.
- Permite múltiples sesiones simultáneas de administración.
- Permite administración vía web y CLI, para administrar vía interface gráfica y línea de comandos mediante puerto consola y protocolo SSHv2.
- Incluye SNMPv1, v2c y v3.
- Soporte de protocolos RCP y SCP
- Protocolos de enrutamiento IPv4 estático y dinámico (RIP, OSPF, BGP).
- Registro de eventos vía Syslog.
- Protocolos SNTP, DHCP y DNS.

- Funcionalidad de "puerto espejo" por puerto o grupo de puertos y por VLAN.
- Creación de canales Ethernet que cumplan con IEEE 802.3ad, mediante el protocolo LACP, que permiten usar cualquier puerto del mismo tipo y velocidad.
- Soporte de múltiples instancias virtuales para segmentar el sistema operativo y los recursos de hardware para emular Switches virtuales.
- IEEE 802.1D Spanning Tree Protocol.
- IEEE 802.1s Multiple Spanning Tree Protocol.
- IEEE 802.1w Rapid Spanning Tree Protocol.
- IEEE 802.1ab LLDP. IEEE 802.1ae MAC Security (Link-Layer cryptography).
- IEEE 802.3ab 1000BASE-T, Gigabit sobre cobre
- IEEE 802.3z Gigabit Ethernet sobre fibra.
- IEEE 802.3ae 10 Gigabit Ethernet.
- IEEE 802.10 VLAN Tagging.
- IEEE 802.1p Class-of-Service (CoS) Tagging for Ethernet frames.
- IEEE 802.1x Port-based network access control.

De acuerdo con las especificaciones técnicas mínimas, los dispositivos de comunicaciones de la red telemática presentan la característica de una red con arquitectura tradicional. Los dispositivos son administrados de forma local balo CLI compuesto por un sistema operativo propietario, por interfaces GUI propietarios o por gestores de administración propietarios que utilizan protocolos como SNMP para su gestión.

Los protocolos de enrutamiento usado en la Red Telemática tales como BGP, OSPF, RIP v1/v2 y rutas estáticas no tienen en cuenta las demandas de trafico que existen en la red en tiempo real dificultando la realización de pruebas con nuevos protocolos, cambios de tecnología y de nuevas herramientas de administración, debido a que existe la posibilidad de presentarse condiciones no deseas que pongan en riesgo la estabilidad de la red. Los protocolos de enrutamiento tradicionales conocen el ancho de banda y costo de los enlaces directamente conectados a ellos, pero no tienen un conocimiento global de la red y todos sus enlaces, además cada Router tiene su propio algoritmo de enrutamiento, lo que hace que el control sea descentralizado, y no se pueda predecir el comportamiento exacto de la red completa.

Los protocolos estandarizados como IEEE pueden ser ejecutados dentro de una red SDN mediante los distintos módulos que nos ofrecen el controlador Opendaylight. Las tablas de flujos instalados por el controlador Opendayligh hacia los switches Openflow reemplazaran a las rutas obtenidas por los protocolos de enrutamiento tradicionales.

Las técnicas de gestión de los dispositivos tradicionales en la red telemática son las siguientes:

- Interface de línea de comandos (CLI)
- Gestores propietarios – SNMP

3.2.1. INTERFAZ DE LÍNEA DE COMANDOS (CLI)

La red telemática de la UNMSM utiliza la interfaz de línea de comandos para la gestión de los dispositivos de red. CLI permite a los administradores de red dar instrucciones a los dispositivos de red mediante líneas de texto simple.

CLI consiste en un "prompt" donde se escriben las órdenes al dispositivo, el usuario ejecuta una orden para pasar a la siguiente línea. El sistema operativo del dispositivo de red interpreta la orden de acuerdo con una sintaxis preestablecida por el fabricante.

Según **Big Switch Inc (2011),** la configuración de los dispositivos de red presenta los siguientes inconvenientes cuando se utilizan CLI para su configuración:

1) CLI está destinada principalmente para ser utilizada para configurar dispositivos de red por personas, CLI no fue diseñada para que capas de software encima de ella puedan programar la red.

2) CLI resulta difícil mantener actualizado un modelo centralizado de las configuraciones de los dispositivos de red porque las redes LAN tienen diferentes dispositivos de red con diferentes versiones de CLI en constante cambio.

3) La configuración a través de CLI tiene limitaciones en su funcionalidad debido a los protocolos existentes. CLI no expone las capacidades del dispositivo de red de manera flexible.

Según **Big Switch Inc (2011),** existen protocolos, sistemas de administración, etc. que intentan evitar la fragilidad y la dificultad de la configuración a través de CLI, pero tienen algunos de los mismos problemas fundamentales que CLI, principalmente en el tercer punto. Sin embargo, CLI puede usarse para depurar, solucionar problemas o interactuar en tiempo real con el dispositivo

La red LAN de la red telemática cuenta con dispositivos de red de los fabricantes como Cisco, Fortinet, Checkpoint, entre otros, y la gestión de estos dispositivos es a nivel local generalmente mediante CLI. Los distintos fabricantes de dispositivos de red implemente su propia sintaxis de línea de comandos e incluso cuentan con certificaciones de especialización para la administración de sus dispositivos, incrementando la dificultad de la integración y gestión de la red.

En la tabla 3.1, se observa las diferencias de las sintaxis de algunos de los principales fabricantes de dispositivos de red.

Fabricante	CISCO	JUNIPER	NOKIA	HUAWEI
Sistema Operativo	IOS	JUNOS	TiMOS	HVRP
COMANDOS PARA VISUALIZAR PARÁMETROS ESPECÍFICOS DE PROTOCOLOS Y/O SISTEMA	show	show	show	display
	exit	exit/up	exit	quit
	running	do	---	---
	reload	request system reboot	admin reboot now	reboot
	show running-config	show configuration	admin display-config	display current-configuration
	show ntp status	show ntp status	show system ntp	display ntp-service status
	show platform	show chassis fpc	show car, show mda	display device pic-status
	show processes cpu	show system processes extensive	show system cpu	display cpu-usage
	show ospf summary	show ospf overview	show router ospf status	display ospf brief
	show bgp	show route protocol bgp	show router bgp routes	display bgp routing-table
	show mpls ldp summary	show ldp overview	show mpls ldp summary	display mpls ldp all

Tabla 3.1: Tabla comparativa de comandos CLI por fabricante.
Fuente: Elaboración propia.

Los diferentes tipos de sintaxis y opciones que ofrecen el CLI de cada fabricante complica la gestión de la red. Las fabricantes de los dispositivos de red incluyen cursos de capacitación y entrenamiento para la correcta operación de sus dispositivos. Las certificaciones propietarias ofrecidas se encuentran en la tabla 3.2, generan una inversión de gasto operacional (Opex) y exigen personal altamente capacitado experto en configuración de los dispositivos de red.

Según **MAREK (2016),** cuando la compañía de telecomunicaciones AT&T despliegue su red SDN a un 75%, los gastos operacionales de su red se reducida hasta un 40 por ciento o 50 por ciento. Ese ahorro de costos ocurrirá porque muchas de las operaciones manuales serán reemplazadas por scripts y procedimientos automatizados.

	Fabricante			
Nivel de certificación	**CISCO**	**JUNIPER**	**NOKIA**	**HUAWEI**
Asociado	CCNA	JNCIA & JNCIS	NRS I	HCNA
Profesional	CCNP	JNCIP	NRS II	HCNP
Experto	CCIE	JNCIE	SRA	HCIE

Tabla 3.2: Tabla comparativa de certificaciones del fabricante.
Fuente: Elaboración propia

Según **GARTNER (2016),** para el año 2020 solo el 30% de los equipos de operaciones de red usarán la CLI como su interfaz principal debido a las siguientes causas:

- Las organizaciones necesitaran aumentar la agilidad de la red mediante la automatización tareas operacionales; por ejemplo, reducir la dependencia de los servicios manuales en los dispositivos de red para cambios de configuración repetitivos.
- Las redes empresariales LAN están buscando soluciones para operar partes de la red como un todo, desde un solo punto de control, y no dispositivo por dispositivo.
- CLI se asocia con cambios de configuración manual, que son la causa principal de interrupciones de red en la empresa.
- Las nuevas soluciones de red aprovechan una API para permitir la integración con otras herramientas de orquestación de infraestructura y la automatización de tareas repetitivas.

Según **GARTNER (2016),** la reducción de la dependencia del CLI pueden facilitar el cambio de dispositivos de redes, ya que reduce el tiempo y los costos asociados. SDN brinda a las empresas una gama más amplia de opciones para la selección del hardware de la red.

La Red Telemática, bajo una arquitectura basada en SDN, puede alejarse de las operaciones a través de CLI y utilizar las APIs de programación para lograr una mayor agilidad en los proyectos de modernización y automatización de la red

3.2.2. GESTORES PROPIETARIOS - SNMP

El protocolo simple de gestión de red (SNMP) es un protocolo estándar de Internet Engineering Task Force (IETF) para la supervisión y gestión de sistemas y dispositivos en una red. Las funciones soportadas por el protocolo SNMP son la solicitud y recuperación de datos, el establecimiento o la escritura de datos y condiciones de excepción que indican la aparición de sucesos.

Según **CASE ed al. (2002)**, SNMP permite a una aplicación de gestión consultar información de un dispositivo gestionado a través una Management Information Base (MIB) estructurada de forma jerárquica para definir el significado y el tipo de un valor determinado. El dispositivo gestionado se denomina agente SNMP y permite el envío y recepción de información SNMP.

Un gestor de SNMP recibe la información de los agentes SNMP para supervisar un dispositivo en intervalos de tiempo. El gestor de SNMP también puede cambiar la configuración del dispositivo de red estableciendo determinados valores.

La Red Telemática utiliza el protocolo SNMP para gestionar y monitorear los dispositivos de red. SNMP también permite incluye utilidades graficas de software libre como MRTG (Multi Router Traffic Grapher), CACTI, entre otros; que se utiliza para representar gráficamente los datos de los gestores SNMP a través de páginas HTML, entre algunas de la información que puede graficar es el tráfico entrante y saliente de las interfaces de red (ver figura 3.2),

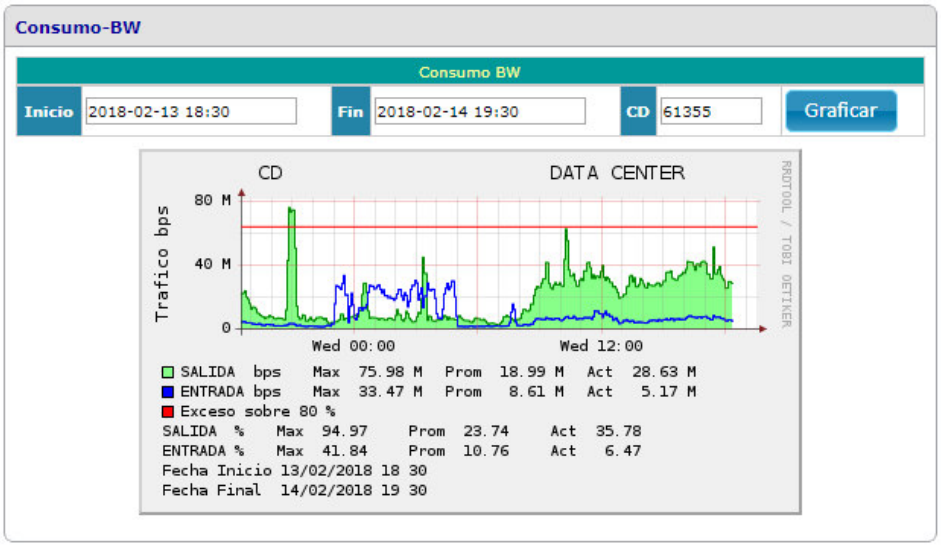

Figura 3.2: Grafica de tráfico entrante y saliente por SNMP.
Fuente: Elaboración propia.

Sin embargo, SNMP tiene limitación en la cantidad de tipos de datos que puede manejar, funcionalidad reducida y no facilita el diseño de MIBs propios, por lo cual están bajo el control de los distintos dispositivos de red y fabricantes.

3.3. TOPOLOGÍA DE LA RED TELEMÁTICA Y CAMPUS DE LA UNMSM

Según la presentación de la situación actual y plan de trabajo presentado en el año 2016 por la oficina de la Red Telemática de San Marcos, la red telemática fue pasando por distintas etapas desde 1996 hasta la IV etapa correspondiente a los años 2012 – 2016. En la IV etapa se tiene un aproximado de 83 switches del proveedor Cisco proveyendo conectividad y servicios a 6000 computadores aproximadamente.

En la figura 3.3 se muestra la topología de red de la Red Telemática y Campus de la UNMSM adaptada de la presentación de la situación actual y plan de trabajo presentado en el año 2016:

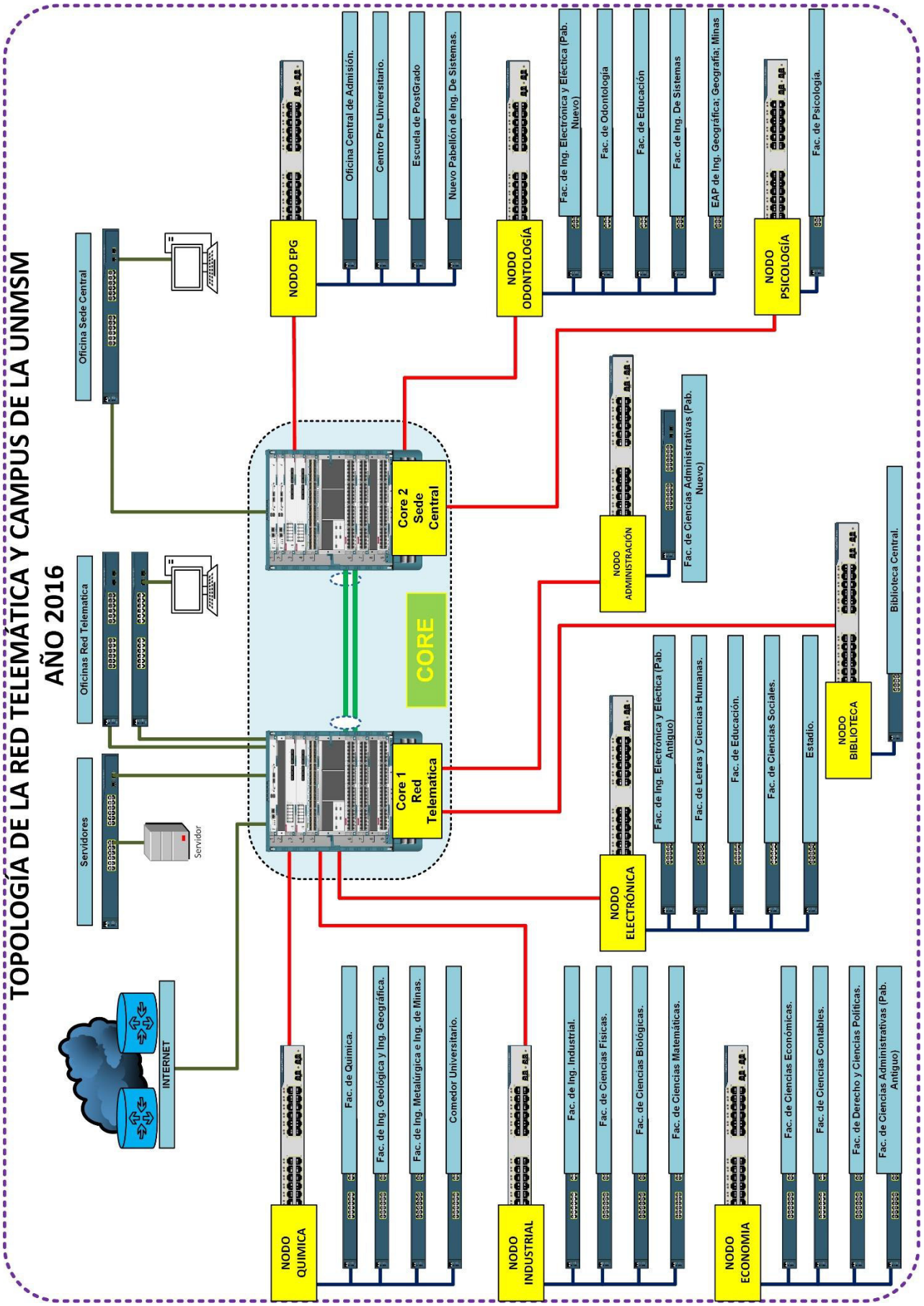

Figura 3.3: Topología de la Red Telemática y Campus de la UNMSM.
Fuente: Elaboración propia basado en la situación actual de la red del año 2016.

Según **KREUTZ (2015),** la posibilidad de establecer parámetros de configuración en una gran cantidad de dispositivos de red usando métodos como CLI y SNMP pueden ser engorrosos y difíciles de mantener. Además, están orientados a tareas de gestión estáticas y no a tareas dinámicas, frecuentes y automatizadas requeridas en entornos tales como las grandes redes actuales.

3.4.CASOS DE USO

3.4.1. AT&T

Según **DANO (2016**), Fuetsch (ejecutivo de la empresa proveedora de servicios AT&T) explico que la razón por la cual los gastos de capital de AT & T disminuyeron fueron debido al movimiento del operador hacia redes definidas por software (SDN) y virtualización de funciones de red (NFV). La operadora se encuentra en camino de tener el 30% de su red virtualizado y tiene el objetivo de virtualizar el 75% de su red para el 2020. Para la virtualización de su red, la operadora traslado las funciones de red de costosos dispositivos de hardware hacia software que se ejecuta en un hardware básico menos costoso. Los cuatro operadores inalámbricos más importantes de los EE. UU. gastaron un total combinado de $ 7,300 millones en el segundo trimestre de 2016, según Jennifer Fritzsche de Wells Fargo Securities, un 10 por ciento menos que los $ 8,1 mil millones que gastaron durante el mismo período hace un año

Según **DANO (2016**), Fuetsch agregó que el movimiento continuo de AT & T hacia las tecnologías SDN y NFV también ayudaron al operador a reducir sus gastos operativos. Agrego que ya no es necesario contar con técnicos para instalar nuevas soluciones de red, con SDN es automático. Indico que debido a que las funciones de red están virtualizadas, es posible mover y cambiar la capacidad en una red en producción.

3.4.2. GOOGLE

La empresa de servicios Google Inc. implemento una red WAN SDN para proporcionar escalabilidad (múltiples terabits de ancho de banda) y tolerancia a fallas. Los dispositivos de red se conectan entre sí a múltiples controladores OpenFlow. Los dispositivos de red fueron fabricados por la misma compañía

Según **Big Switch (2012),** los múltiples controladores Openflow minimizan los puntos de falla. El controlador Openflow recopila los datos de topología, métricas utilizadas en tiempo real de la red y la demanda del ancho de banda de las aplicaciones y servicios. Con esta información, la aplicación SDN calculan las asignaciones de ruta para los flujos de tráfico y luego programa las rutas en los switches utilizando OpenFlow. En el caso de cambios de demanda de ancho de banda o eventos de red, la aplicación SDN vuelve a calcular las asignaciones de ruta y reprograma los switches.

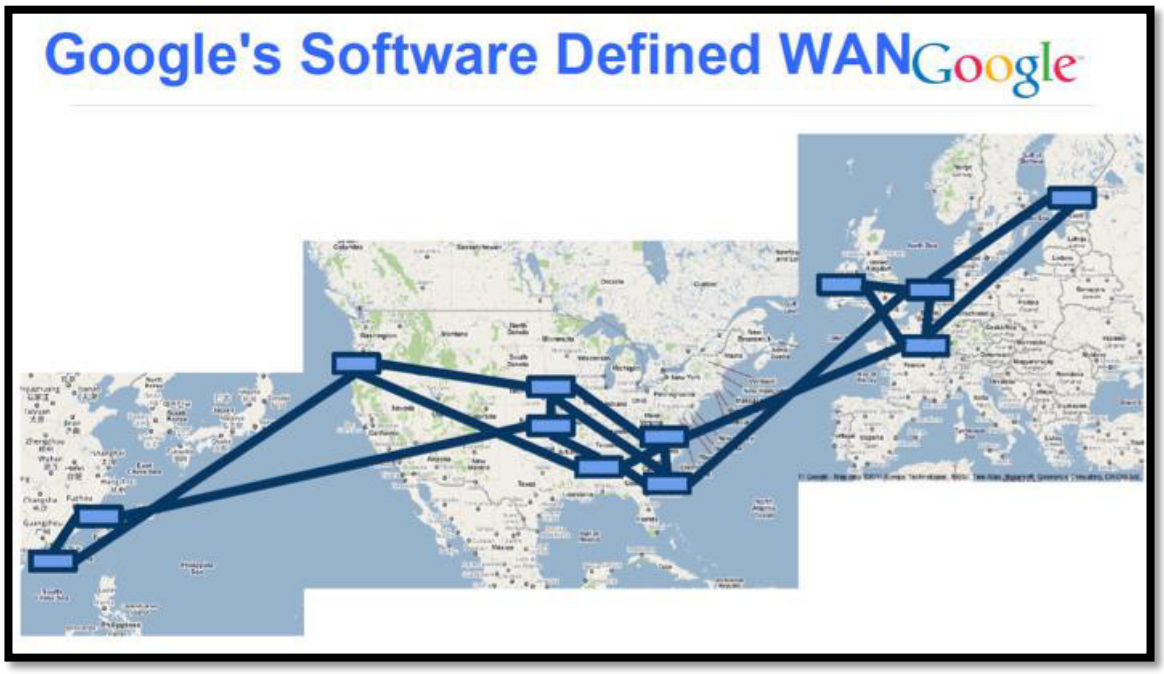

Figura 3.4: Red WAN SDN de Google.
Fuente: Big Switch (2012)

Las motivaciones de Google para implementar SDN fueron que el rápido crecimiento del ancho de banda de Internet llevó a la empresa a pensar que no podrían mantenerse sus niveles de escalabilidad, tolerancia a fallos, control y costes con tecnologías de WAN tradicionales.

Según **Big Switch (2012),** Google observo una convergencia más rápida con su red y una mayor confiabilidad, concluyendo que la convergencia calculada centralmente por SDN era buena y precisa. Estas características permitieron que la red de Google funcione con una ratio de utilización cercana al 100% y de media un 70% en largos períodos de tiempo, ratio que se corresponde con un incremento de la eficiencia de dos a tres veces, comparándola con las tecnologías de gestión estándares.

Sin embargo, la red Google presento inconvenientes. Por ejemplo, presento "bottlenecks" o cuellos de botella, especialmente al enviar paquetes del plano de control al plano de datos. Los cuellos de botella experimentados estuvieron ligados al rendimiento o capacidad del controlador para manejar todos los paquetes del plano de control en un mismo instante de tiempo.

Del caso de Google se pueden extrapolar ideas a un gran número de implementaciones de SDN en la vida real. A pesar de las particularidades de su red interna hay una serie de prácticas a considerar. Una de ellas es la aproximación híbrida, que consiste en que la red soporte los protocolos tradicionales a la vez que Openflow, de forma que la transición no es traumática y elimina suspicacias a la hora de adoptar SDN.

CAPITULO IV

DISEÑO DE LA RED TELEMÁTICA BAJO LA ARQUITECTURA SDN

4.1.ENTORNO DE SIMULACIÓN

Para la presente simulación se utilizó una laptop con las siguientes características:

- CPU: Intel Core i7
- Memoria RAM: 12 GB
- Tarjeta de video dedicada: NVIDIA 4 GB
- Sistema Operativo: OS Linux Ubuntu 17.04

Cada máquina virtual de VirtualBox tiene las siguientes características mostradas en la tabla 4.1.

Máquina Virtual	CPU	Memoria RAM	Memoria de almacenamiento	Sistema Operativo	Versión
Mininet	2 cores	4 MB	40 GB	Linux Ubuntu 14.06	V2.2.2 con Openflow 1.3
Opendaylight	2 cores	8 MB	40 GB	Linux Ubuntu 14.06	Nitrogen SR1

Tabla 4.1: Características de las máquinas virtuales.
Fuente: Elaboración propia.

Asimismo, se usaron los softwares libres Mininet, GNS3 y el controlador Opendaylight para la simulación de la red definida por software.

4.2.ESQUEMA GENERAL DEL ENTORNO DE SIMULACIÓN

El esquema general del entorno de simulación se muestra en la figura 4.1. Las máquinas virtuales de Mininet y Opendaylight se ejecutaron en instancias distintas. Se utilizo el software GNS3 para proveer conectividad entre ambas máquinas virtuales dentro de una misma área de red local. Los procesos que se ejecutaran en los planos de datos, control y aplicación de la red SDN son los siguientes:

- Plano de datos:

Mininet ejecuta switch virtuales con sistema operativo Openvswitch. Openvswitch permite ejecutar los procesos necesarios para la conmutación de paquetes y es compatible con características tales como Openflow, LACP, VLAN 802.1q, etc.

- Plano de control y de aplicación:

Opendaylight es un controlador SDN que administrara la red simulada en Mininet a través de Openflow y ejecutara las aplicaciones que permitirán la conmutación de paquetes de origen a destino, entre otras aplicaciones.

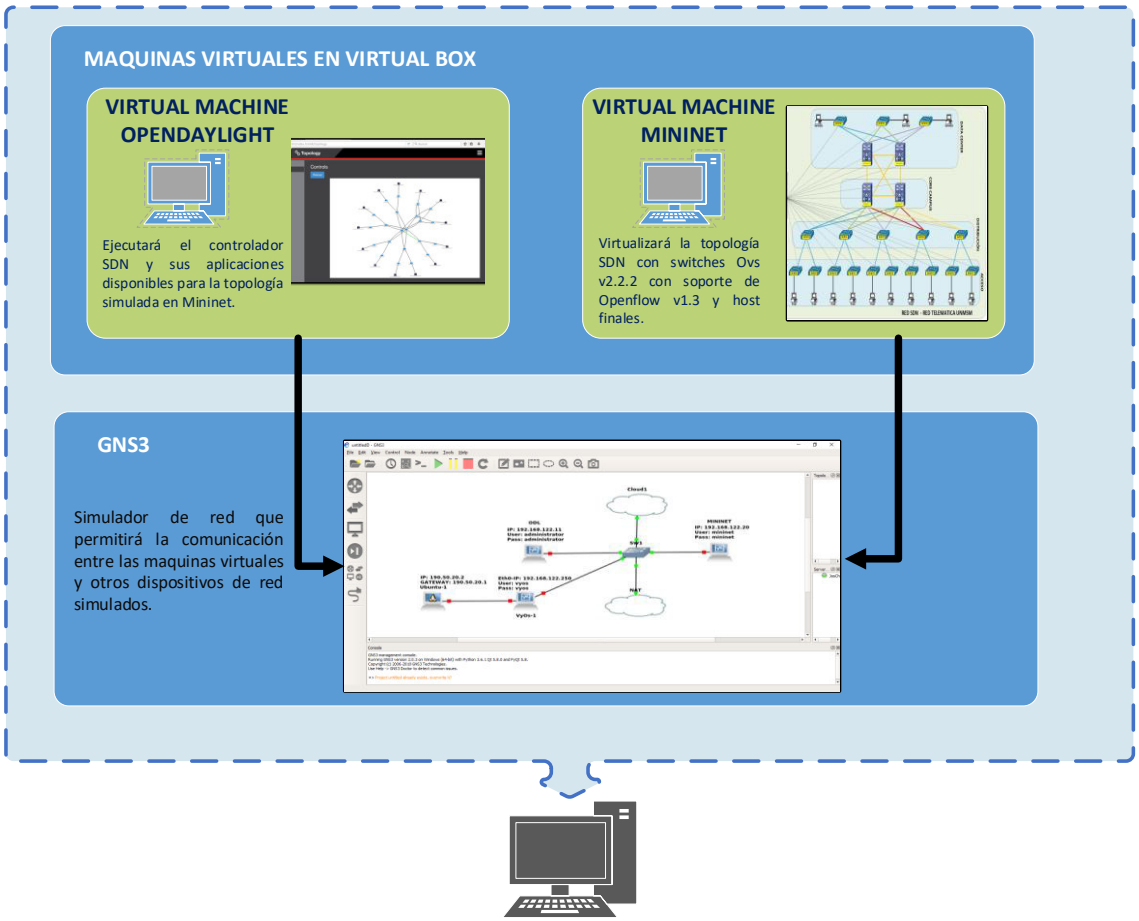

Figura 4.1: Esquema general de simulación.
Fuente: Elaboración propia

A continuación, se describen los softwares libres Mininet, GNS3 y el controlador Opendaylight utilizados en la simulación:

4.2.1. MININET

Mininet es un sistema que permite la creación de redes virtuales, virtualizando los kernel, switch y códigos de aplicación, en una sola computadora a través de máquinas virtuales, virtualización en la nube o de forma nativa. Mininet crea redes definidas por software utilizando mecanismos de virtualización livianos, estas características permiten a Mininet crear, interactuar, personalizar y compartir rápidamente los prototipos de red implementados.

Según **LEAO ed al. (2014),** algunas características de Mininet son:

1) Flexibilidad: Es posible establecer nuevas topologías y funciones de software usando lenguajes de programación y sistemas operativos comunes.

2) Aplicabilidad: Las implementaciones utilizadas en la topología de red simulada son funcionales en redes reales sin necesidad de cambiar el código fuente de las aplicaciones.
3) Tiempo real: Las interactividad, gestión y ejecución de cambios en la red simulada son en tiempo real como si ocurrieran en redes reales.
4) Escalabilidad: El entorno de creación de prototipos puede escalar a redes grandes con cientos o miles de switch en una sola computadora, dependiendo de los recursos de hardware de la computadora.
5) Realista: El comportamiento de la red simulada representa el comportamiento en tiempo real con un alto grado de confianza.
6) Simulaciones compartibles: Las simulaciones creadas pueden ser compartidas fácilmente con otros colaboradores para su ejecución y modificaciones.

Mininet puede crear dispositivos de red que permiten la integración con la arquitectura SDN. Estos dispositivos de red incluyen hosts, switches, controladores y enlaces. Un host en Mininet es un proceso simple con su propio entorno de red ejecutándose en el sistema operativo. Cada uno proporciona procesos con interfaz de red virtual de propiedad independiente para cada dispositivo de red, incluyendo puertos, direcciones y tablas de enrutamiento (como ARP e IP). Los switches Openflow creados por Mininet proporcionan la misma semántica de entrega de paquetes que proporcionaría un switch real. En la simulación de Mininet, los controladores se pueden ejecutar en la red real o simulada siempre que la máquina en la que se ejecutan los switch tenga conectividad con el controlador.

En la presente simulación, se instaló Mininet 2.2.2 en una máquina virtual utilizando el software de virtualización llamado Virtual Box, y se utilizó la versión 1.3 de Openflow.

4.2.2. CONTROLADOR OPENDAYLIGHT

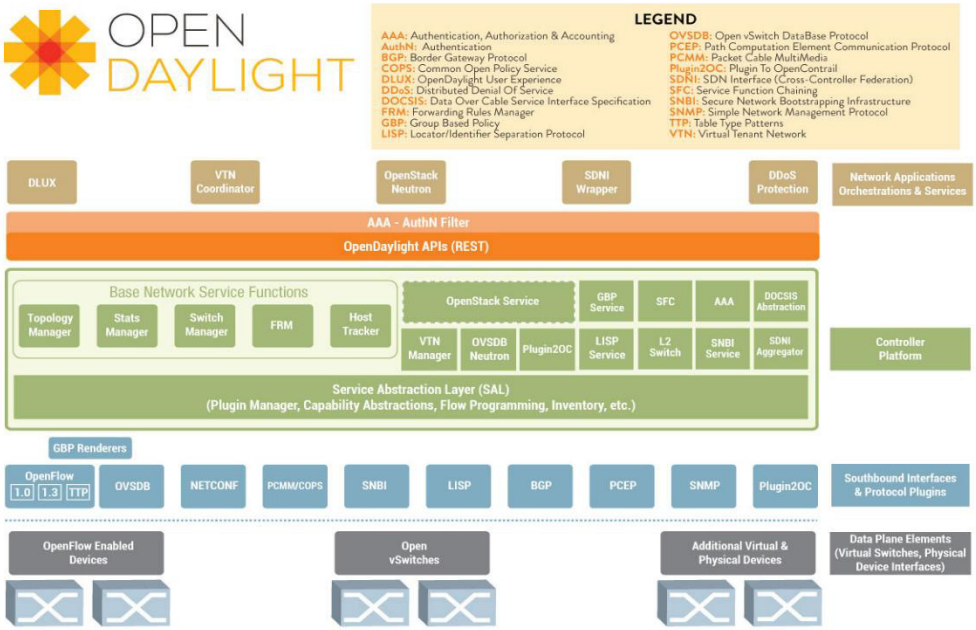

Figura 4.2: Arquitectura del controlador Opendaylight.
Fuente: SDN HUB (2015).

Según **OPENDAYLIGHT (2017)**, OpenDaylight es un proyecto de código abierto compatible con IBM, Cisco, Juniper, VMWare y otros proveedores importantes. OpenDaylight es un controlador SDN implementada con lenguaje Java y se puede instalar en cualquier sistema operativo que admita Java.

Como se observa en la figura 4.2, la arquitectura de Opendaylight SDN Controller tiene varias capas. La capa superior consiste en aplicaciones de gestión y lógicas de la red. La capa intermedia es la capa de infraestructura y la capa inferior consiste en dispositivos físicos y virtuales (switch).

Según **OPENDAYLIGHT (2017)**, la capa intermedia es el marco en el que se pueden manifestar las abstracciones de SDN, esta capa aloja las north-bound y south-bound API. Opendaylight admite el marco REST bidireccional para la north-bound API. Las aplicaciones, las cuales están por encima de la capa intermedia, utilizan el controlador para recopilar la información de la red, ejecutar algoritmos para realizar análisis y usar esta información para realizar cambios en la red a través del controlador. La interfaz south-bound API puede soportar múltiples protocolos como Openflow y BGP-LS como complementos separados.

	CONTROLADORES					
CASOS DE USO	Trema	Nox/Pox	RYU	Floodlight	Opendaylight	ONOS
Virtualización de la red con Virtual Overlays.	SI	SI	SI	PARCIAL	SI	NO
Virtualización de red Hop-by-Hop.	NO	NO	NO	SI	SI	YES
Soporta OpenStack Neutron.	NO	NO	SI	SI	SI	NO
Interoperabilidad con redes tradicionales.	NO	NO	NO	NO	SI	PARCIAL
L4-L7 Servicios de inserción y servicios de chaining.	NO	NO	PARCIAL	NO	SI	PARCIAL
Monitoreo de la red.	PARCIAL	PARCIAL	SI	SI	SI	YES
Políticas de acción.	NO	NO	NO	PARCIAL	SI	PARCIAL
Balanceo de cargas.	NO	NO	NO	NO	SI	NO
Ingeniería de trafico.	PARCIAL	PARCIAL	PARCIAL	PARCIAL	SI	PARCIAL
TAPS dinámicos.	NO	NO	SI	SI	SI	NO
Optimización de red Multi-Layer.	NO	NO	NO	NO	PARCIAL	PARCIAL
Red de campus.	PARCIAL	PARCIAL	PARCIAL	PARCIAL	PARCIAL	NO
Enrutamiento.	SI	SI	SI	SI	SI	SI

Tabla 4.2: Tabla comparativa de controladores SDN según caso de uso.
Fuente: RAO (2015).

Como se puede observar en la tabla 4.2, Opendaylight nos ofrece más características de usos que los otros controladores de software libre, además de tener soporte y una gran comunidad de desarrollo. Esto permite que Opendaylight sea de inmensa popularidad en la comunidad de software libre para realizar test en ambientes de simulación.

Por estos motivos se utilizó el controlador Opendaylight para la simulación de la red SDN, se utilizó el reléase Nitrogen SR1 de Opendaylight en una máquina virtual utilizando el software de virtualización llamado Virtual Box.

4.2.3. GNS3

GNS3 es un software libre de simulación gráfico de red que permite diseñar topologías de red complejas, así como permitir utilizar maquina virtuales a través que Qemu y VirtualBox.

Según **GNS3 (2017),** el software permite crear, diseñar y testear una red en un entorno virtual libre de riesgos además de tener una gran comunidad de ayuda y soporte. GNS3 también brinda lo siguiente.

- Simulación de red en tiempo real para pruebas previas a la implementación sin necesidad de hardware de red.
- Crear mapas de red dinámicos para pruebas de resolución de problemas y prueba de concepto (POC).
- GNS3 permite la conexión con cualquier red real.

Se utiliza GNS3 2.1 en la presente simulación para crear un ambiente de prueba e incluir las maquinas creadas en VirtualBox en una misma red, además de permitir unir nuestra red SDN con dispositivos de red tradicional de manera virtualizada.

4.3. TIPOS DE TRAFICO EN LA RED (BUM)

BUM es el acrónimo de Broadcast, Unknown, Multicast, los cuales son los tipos de tráfico que se manifiestan en la capa 2. En una red tradicional LAN como la Red Telemática, el BUM es generado por diferentes tipos de servicios, por ejemplo, de los protocolos DHCP, ARP, VoIP, etc.

En la red LAN de la Red Telemática, cuando los dispositivos de capa 2 (switch) tradicionales reciben un paquete, se crea una asignación en la tabla TCAM registrando la dirección MAC de origen y el puerto por el cual ingreso del paquete. Si un switch recibe un paquete destinado a una dirección MAC que no existe en su tabla de direcciones TCAM, o es un paquete broadcast (por ejemplo, paquetes con destinado a FF; FF: FF: FF: FF: FF), el switch de capa 2 enviara el paquete por todos sus puertos, excepto el puerto que recibió el paquete broadcast, inundando de esta manera segmentos de red con tráfico innecesario.

En la red SDN propuesta, el controlador SDN conocerá toda la red y los hosts. Por lo cual, el switch Openflow enviará los paquetes broadcast, multicast y unicast que no tengan coincidencia en las tablas de flujo al controlador, y será el controlador SDN quien de la información necesario para tratar al paquete.

Un controlador SDN, con el módulo L2-SWITCH de Opendaylight, ira construyendo las tablas de flujo de los switch de tal forma que se reduzca el número de solicitudes desde el switch al controlador SDN y pueda reducir el tráfico BUM innecesario.

4.4.CONECTIVIDAD HACIA REDES TRADICIONALES

Opendaylight permite el uso del protocolo de enrutamiento Border Gateway Protocol (BGP), el cual está definido en la RFC4271. Según la RFC4271 propuesta por **REKHTER ed al. (2006)**, BGP es un protocolo que nos permite intercambiar información de enrutamiento entre sistemas autónomos (AS), lo cual permite la comunicación a través de distintas organizaciones en el mundo. Para la presente simulación de la red SDN, el protocolo BGP permitirá intercambiar información de enrutamiento entre el dominio SDN con routers externos.

En la red SDN, la red interna no utiliza los protocolos de enrutamiento IGP, sin embargo, es posible la integración de la red interna con redes de proveedores de servicio a través del controlador. BGP Path Computation Element Protocol (PCEP) es un protocolo southbound utilizado entre el controlador SDN y los router de la red exterior para intercambiar las tablas de enrutamiento y reenvío (en un nivel superior). Las extensiones del protocolo BGP soportados por Opendaylight son los siguientes:

- BGP PCEP es una extensión de BGP que tiene soporte en routers específicos.
- BGP Linkstate (LS) es otra extensión de BGP que permite a BGP distribuir la información del estado del enlace de los protocolos de enrutamiento.

BGP-PCEP tiene muchos casos de uso para proveedores de servicios, ya que permite a las redes de los proveedores de servicios ser influenciados por controladores SDN.

4.5.MÓDULOS OPENDAYLIGHT UTILIZADOS

El controlador SDN Opendaylight es una plataforma que ejecuta aplicaciones SDN, ya que, sin estas aplicaciones SDN, la red no podría conmutar ningún paquete. Con la integración de estas aplicaciones de red, el controlador decidirá cómo se reenvía los paquetes dentro de la red interna. Las aplicaciones utilizadas para esta simulación de la red telemática bajo un esquema SDN son las siguientes:

4.5.1. DLUXAPPS

DLUX es la interfaz WEB de Opendaylight y permite la integración con diferentes características de Karaf, por ejemplo, la visualización de los nodos, las estadísticas de la red, la topología, permitir el uso de YANG, entre otros.

- Yang es un lenguaje de modelado de datos que se está adoptando rápidamente para modelar Netconf, un protocolo de gestión de red estandarizado IETF, así como para modelar otras interfaces de datos en OpenDaylight.

OPENDAYLIGHT (2017), proporciona una serie de características diferentes de Karaf (figura 4.3), que puede habilitar y deshabilitar por separado:

- CORE: Es el marco de DLUX que proporciona las funcionalidades básicas, como navegación, autenticación, etc.

- IU de YANG: La IU simple para la interacción con el controlador. Se basa en modelos Yang y presenta un formulario para que los usuarios puedan leer o escribir datos de una manera intuitiva.
- YANGMAN: Es la interfaz de usuario YANG avanzado.
- Visualizador YANG: Proporciona una visualización de los modelos YANG en forma gráfica.
- Nodo: Muestra el inventario de los nodos.
- Topología: Muestra la topología OpenFlow de una manera sencilla.

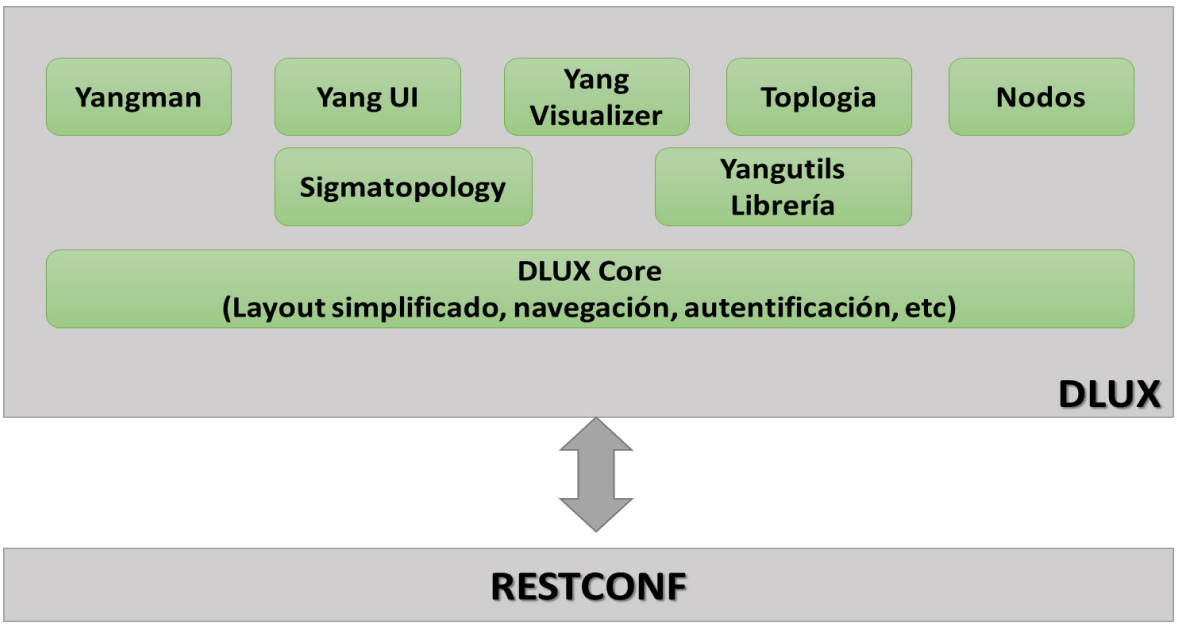

Figura 4.3: Arquitectura del módulo DLUX.
Fuente: OPENDAYLIGHT (2017).

4.5.2. L2 SWITCH

El módulo L2 Switch provee las funcionalidades de conmutación en capa 2, está basada en MD-SAL y permite el aprendizaje sobre cómo se debe reenviar un paquete. El interruptor L2 es una función básica de interruptor de capa 2 dentro de ODL. Incluye manejo específico de capa 2 y también proporciona varios servicios reutilizables. Según **OPENDAYLIGHT (2017)**, L2 Switch tiene la siguiente arquitectura:

- Packet Handler: Proporciona una infraestructura en el controlador ODL para procesar los paquetes entrantes y reenviarlos según sea necesario.
- Loop Remover: Elimina bucles en la red.
- Arp Handler: Handles the decoded ARP packets
- Address Tracker: Aprende las direcciones (MAC e IP) del host de la red
- Host Tracker: Sigue las ubicaciones de los hosts de la red
- L2 Switch Main: Instala las tablas flujos en cada switch según el tráfico de la red

Los paquetes se decodifican, modifican y transmiten como parte de PacketHandler:

- Dirección MAC de host

- ID del switch conectado
- Número de puerto conectado
- VLAN, VNI (VXLAN),
- Etiqueta MPLS.

L2 Switch utiliza esta la información de PacketHandler para informar y usar los siguientes servicios, las cuales se pueden observar en la figura 4.4.

- **Topology Link Data Change Handler**: Proporciona una ruta óptima entre los hosts. utilizando el algoritmo Dijkstra, similar a los algoritmos de enrutamiento IS-IS o OSFP. Path Communication Service aprovecha y modifica los datos de topología, también puede funcionar con múltiples topologías.
- **Flow Writer Service**: Se encarga de programar en todos los swtich intermedios con los flujos adecuados una vez que se ha calculado una ruta óptima entre dos hosts. Realiza un seguimiento del mapeo, desde metaflujo a flujos individuales, de manera que los flujos apropiados se pueden reprogramar cuando un puerto se cae o un switch completo se apaga.
- **STP Service**: Permite el uso del protocolo Spanning Tree para que el controlador ODL participar en el STP de la red. Se utiliza cuando SDN se implementa en conjunto con otra red tradicional y se necesita que los switch Openflow participen dentro del protocolo de STP de los switch tradicionales.

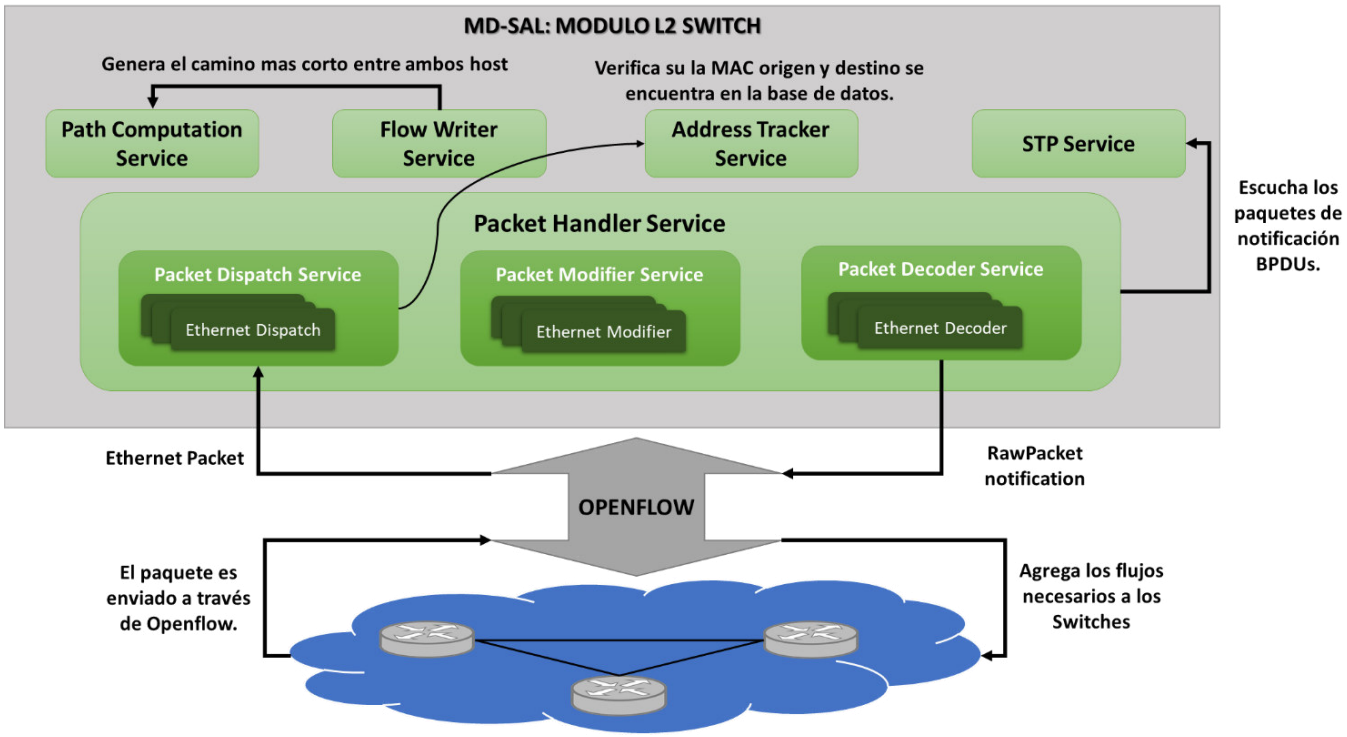

Figura 4.4: Arquitectura del módulo L2 SWITCH.
Fuente: **OPENDAYLIGHT (2017)**.

L2 Switch es un módulo de software que se ejecuta en el servidor Opendaylight y comunica las acciones a través del protocolo Openflow hacia los switches del dominio

SDN. Cuando uno de los switch del dominio SDN recibe un paquete que no tiene ninguna entrada que coincida en sus tables de flujos previamente instalados (origen, destino MAC e IP, etc.), el switch encapsula todo el paquete en un mensaje OpenFlow PACKET_IN y lo envía al controlador Opendaylight. Este paquete se desencapsula en el controlador y se envía al módulo de L2 Switch para que encuentre a donde debe reenviar el paquete y pueda actualizar las tablas de flujo del switch.

El módulo L2 Switch aprende las direcciones MAC de los hosts utilizando los mensajes PACKET_IN del Openflow que recibe de los switches en el dominio SDN, de manera similar a un switch tradicional. Según la documentación de **OPENDAYLIGHT (2017),** el módulo L2 Switch realiza las siguientes acciones:

MAC Origen	MAC Target	Acción del módulo L2 Switch	Comentarios
Desconocido	Desconocido	Aprende la MAC de origen: El paquete broadcast es envía por todos los puertos menos por el cual ingreso.	
Desconocido	Conocido	Aprende la MAC de origen: Paquetes Unicast será enviado al destino.	L2 SWITCH envía el paquete al dispositivo donde se encuentra el host final. El punto de conexión se refiere al host que está unido físicamente al puerto.
Conocido	Desconocido	El paquete broadcast es envía por todos los puertos menos por el cual ingreso.	L2 SWITCH no necesita aprender la MAC de origen ya que conoce. Sin embargo, podría haber un caso en el que un host se haya movido de un nodo a otro en la red, en ese caso se actualizara la MAC.
Conocido	Conocido	L2 SWITCH busca la ruta más corta entre el origen y el destino e instale las entradas de flujos en todos los switch de esta ruta.	

Tabla 4.3: Acciones de L2 SWITCH.
Fuente: OPENDAYLIGHT (2017).

El siguiente diagrama de la figura 4.5, ilustra la comunicación entre los componentes del módulo L2 Switch:

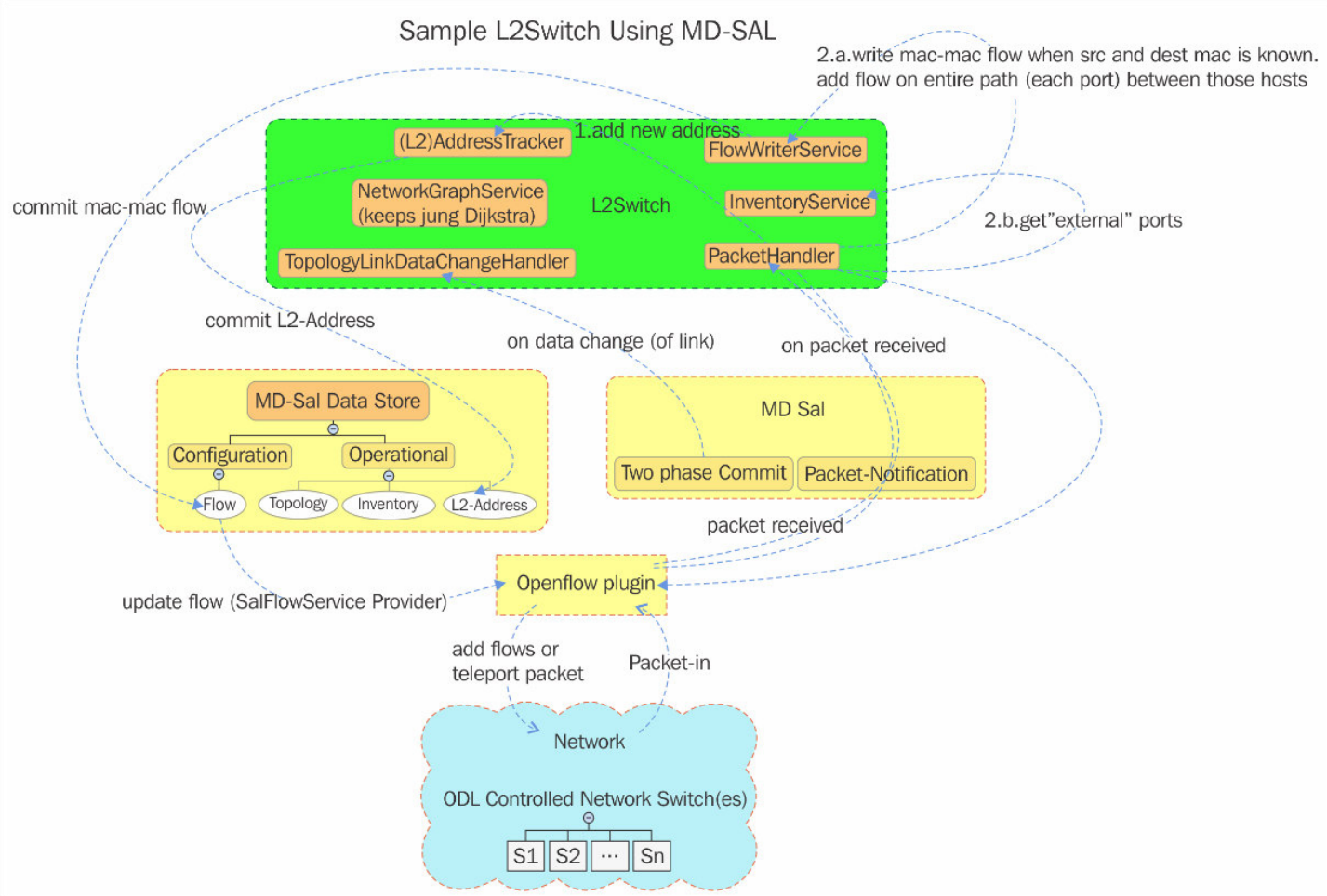

Figura 4.5: Comunicación entre los componentes de L2 SWITCH.
Fuente: OPENDAYLIGHT (2017).

4.5.3. LINK AGGREGATION CONTROL PROTOCOL

Según **OPENDAYLIGHT (2017),** el módulo "Link Aggregation Control Protocol" (LACP) es una implementación del protocolo de control de agregación de enlaces en la forma de un módulo de servicio ML-SAL. Se utiliza para el descubrimiento automático y la agregación de enlaces múltiples entre la red SDN (switches) y otros dispositivos finales (switch o Router) habilitados para LACP.

Según la **IEEE P802.3ad (2017),** el protocolo LACP se lanzó inicialmente como la especificación IEEE Ethernet 802.3ad, pero luego se añadieron más características para su uso en bridge y administrar grupos, como se ve en la especificación 802.1AX.

Los módulos LACP dentro de OpenDaylight reciben y procesan paquetes de control LACP a través del servicio de procesamiento de paquetes SAL. También envían paquetes de control LACP a través de todos los puertos activos del switch utilizando el servicio de procesamiento de paquetes SAL; esto se hace a intervalos regulares, en función de la configuración del temporizador asociado. Como este módulo está diseñado para agregar solo enlaces externos (fuera del dominio SDN), ignora los

paquetes LACP recibidos a través de enlaces internos (dentro de la red SDN). El módulo LACP utiliza el servicio SAL-FLOW para crear LAG dentro del switch siguiendo el diagrama de la figura 4.6.

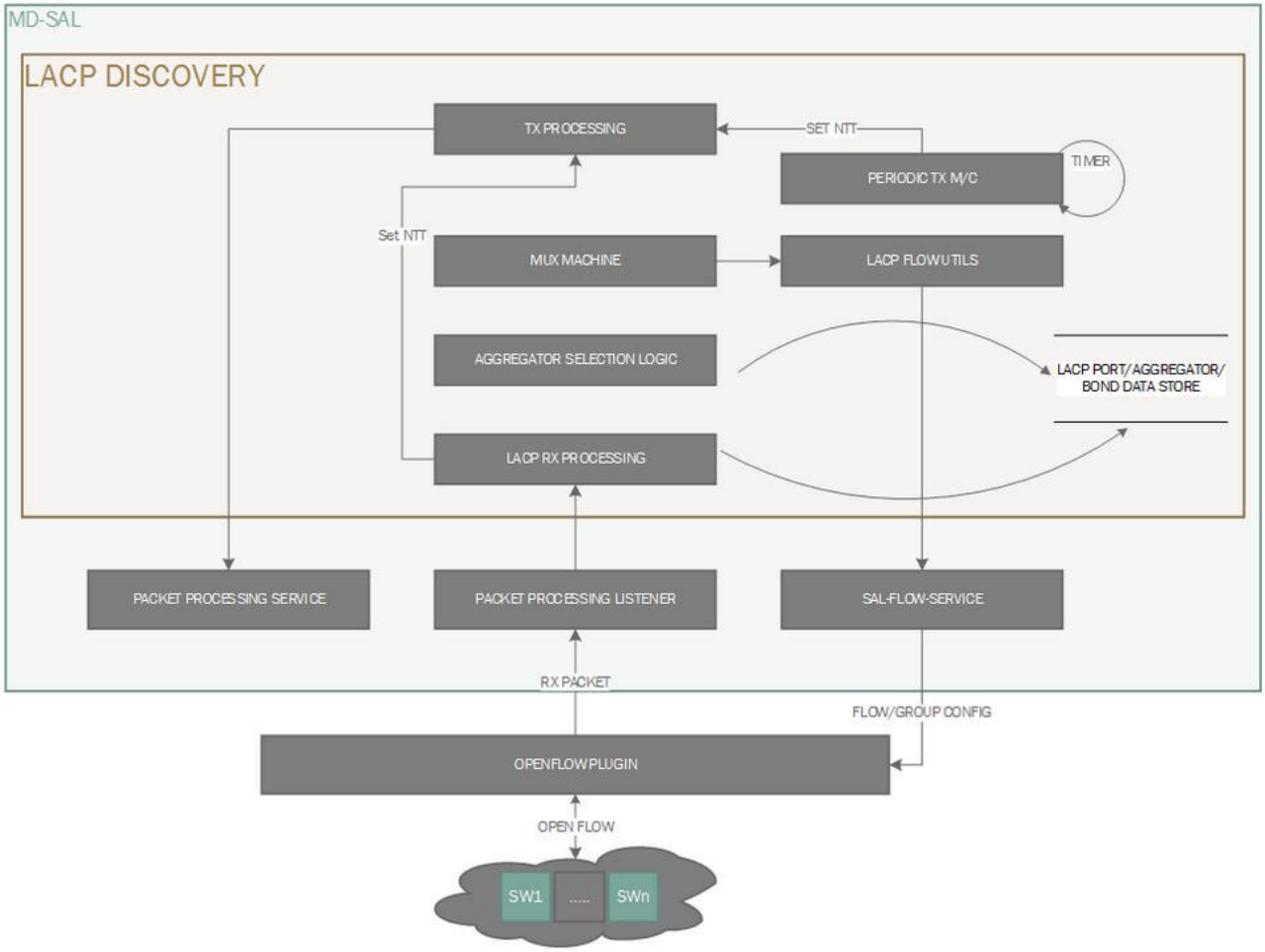

Figura 4.6: Diagrama de los componentes del módulo LACP.
Fuente: OPENDAYLIGHT (2017).

4.5.4. NETCONF: PROTOCOLO SOUTHBOUND

Opendaylight permite el uso del protocolo NETCONF como un protocolo southbound. NETCONF. Según la RFC 6241 propuesta por **ENNS ed al. (2011),** NETCONF provee los mecanismos para instalar, manipular y eliminar las configuraciones de los dispositivos de red. NETCONF utiliza el lenguaje "Extensible Markup Language" (XML) para realizar el envió de la información de configuración a los dispositivos finales. Además, existen APIs que permiten administrar los mensajes NETCONF de una manera sencilla, como al API Postman, utilizado para ambientes de desarrollo.

NETCONF permite interactuar con el controlador Opendaylight además de utilizarlo para comunicarse con los dispositivos físicos de red. Según la RFC 6241 escrita por **ENNS ed al. (2011),** el protocolo NETCONF define las siguientes operaciones.

OPERACIÓN	DESCRIPCIÓN
get	Provee la actual configuración e información del dispositivo.
get-config	Provee toda o parte de la configuración de la base de datos del dispositivo
edit-config	Edita la configuración de la base de datos creando, eliminando o reemplazando contenido.
copy-config	Copia la configuración a la base de datos del dispositivo.
close-session	Solicita la finalización de la sesión NETCONF
kill-session	Fuerza a finalizar la sesión NETCONF.

Tabla 4.4: Acciones de NETCONF.
Fuente: ENNS ed al. (2011)

4.5.5. VIRTUAL TENNAT NETWORK (VTN)

El módulo VTN proporciona soporte de virtualización de red para redes virtuales multiusuario en el controlador Opendaylight.

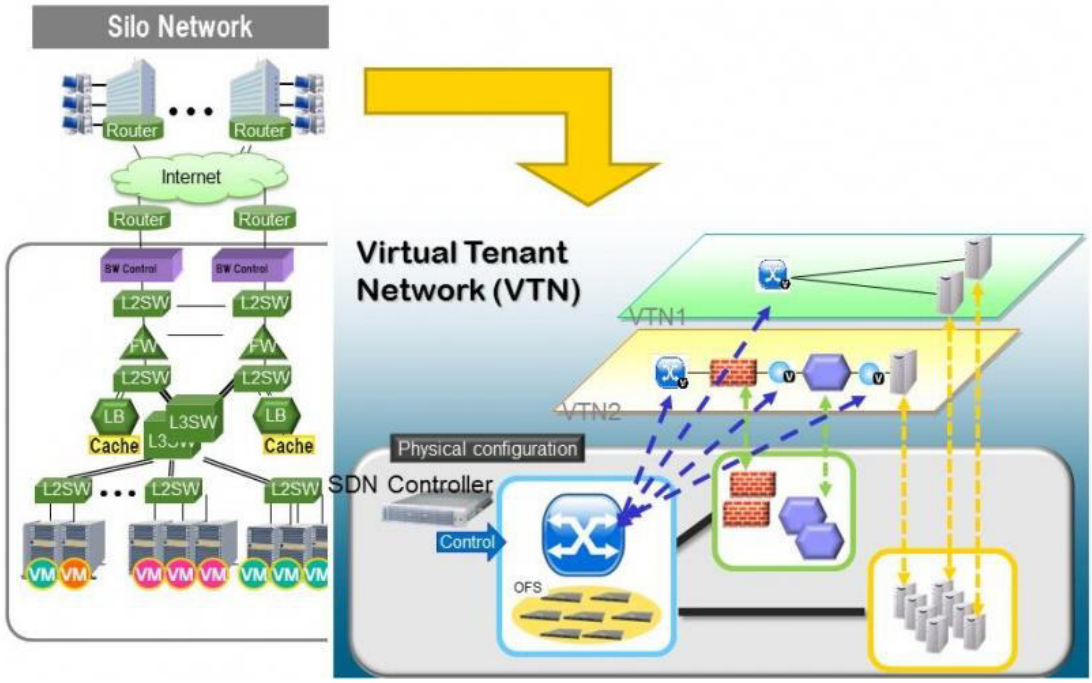

Figura 4.7: Arquitectura del módulo VTP.
Fuente: OPENDAYLIGHT (2017).

Según **OPENDAYLIGHT (2017),** VTN permite crear instancias virtuales donde cada instancia puede tener su propia estructura, tales como router virtuales, switch virtual,

etc. La aplicación principal para VTN son las plataformas privadas de orquestación de nubes, como OpenStack o VMware vSphere.

VTN proporciona una abstracción lógica completa para el usuario. Cuando se crean múltiples redes y enrutadores virtuales en una instancia, VTN se ocupa de todas las comunicaciones subyacentes. Además, admite la separación completamente el plano lógico de los planos físicos y de datos, permitiendo crear un diseño e implementar cualquier red deseada sin tener en cuenta la topología de la red física o las restricciones de ancho de banda.

4.6.DIAGRAMA DEL ENTORNO DE SIMULACIÓN

El diagrama el entorno de simulación en GNS3 se presenta en la figura 4.8.

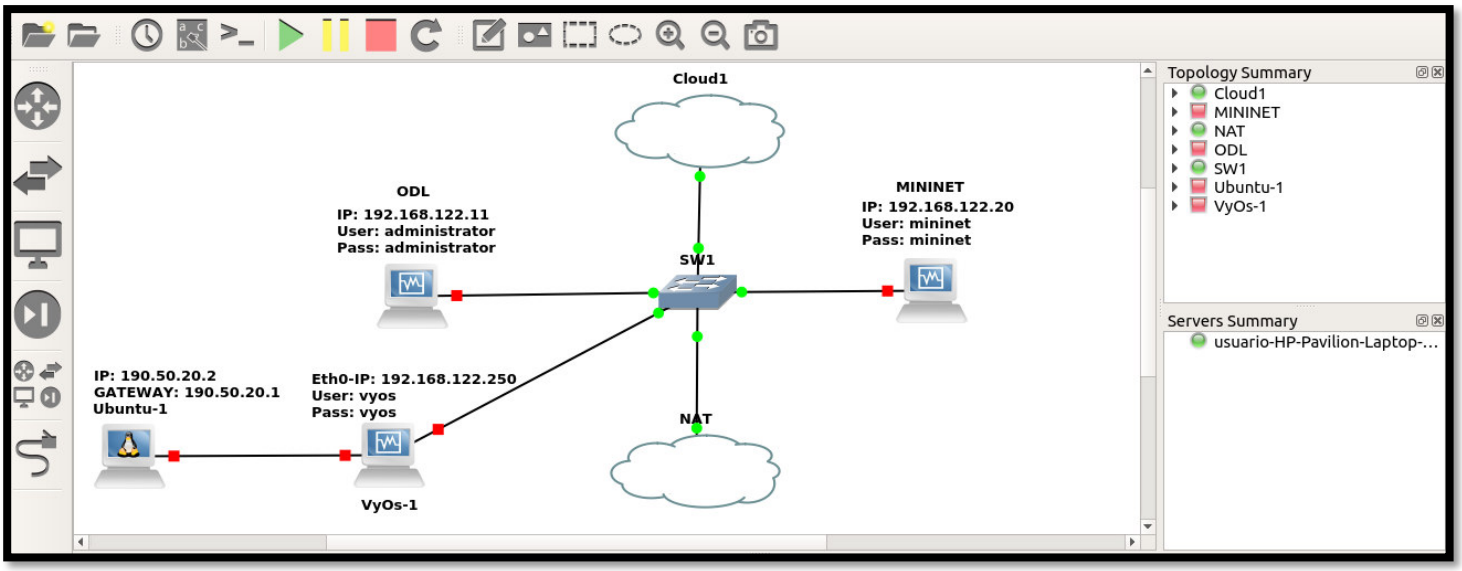

Figura 4.8: Diagrama del entorno de simulación en GNS3.
Fuente: Elaboración propia.

Se utilizó el software GNS3 para simular una red externa y proveer conectividad entre el controlador Opendaylight, la máquina virtual Mininet y otra red externa detrás de un dispositivo Router tradicional. Se utilizaron las siguientes direcciones IP (ver Tabla 4.5):

VM	IP	MASCARA	GATEWAY
OPENDAYLIGHT	192.168.122.11	255.255.255.0	
MININET	192.168.122.20	255.255.255.0	
ROUTER	192.168.122.250	255.255.255.0	
HOST DE LA RED EXTERNA	190.50.20.2	255.255.255.0	190.50.20.1

Tabla 4.5: Direcciones IP de los hosts.
Fuente: Elaboración propia.

La máquina virtual que contiene el controlador Opendaylight se comunicara con la topología SDN a través de la máquina virtual que contiene a Mininet. Adicionalmente, el

controlador SDN se comunicará a través del protocolo BGP con el Router tradicional SO VyOS para intercambiar información con redes externas tradicionales.

Opendaylight orquestará los distintos módulos mencionados anteriormente e instalará las tablas de flujos a los switches SDN instalados dentro de Mininet. El appliance Cloud1 nos permitirá interactuar con la red del entorno de simulación y el appliance NAT proveerá de acceso a internet al entorno de simulación.

4.7. TOPOLOGÍA SIMULADA BASADA EN LA RED TELEMÁTICA

La topología simulada sigue la infraestructura de la Red Telemática y Campus de la UNMSM, el cual está constituido por los siguientes submódulos (ver figura 4.9):

- Compus Core
- Switches de distribución y agregación
- Acceso
- Grande de servidores/Data Center

La capa de Campus Core proporcionara una red conmutada de alta velocidad entre los accesos a los servidores y redes externas, los dispositivos deberán brindar conectividad redundante y de convergencia rápida. La capa de distribución del campus universitario agrega todos los switches de acceso, en esta capa se puede implementar QoS, redundancia y balanceo de carga. Los switches de acceso del campus universitario proporcionan acceso a los usuarios finales.

La granja de servidores o Data Center proporciona acceso de alta velocidad y alta disponibilidad (redundancia) a los servidores, tales como, servidores de datos, servidores web, servidores de aplicaciones, servidores de correo electrónico, etc.

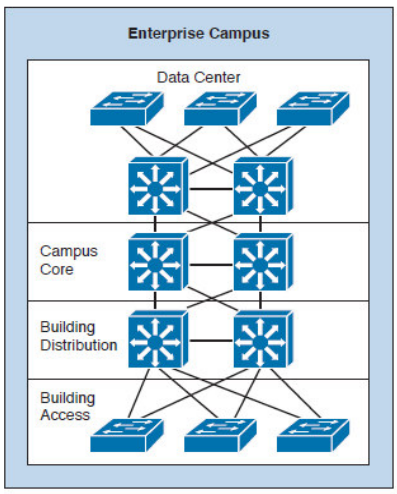

Figura 4.9: Módulos de la Red Campus.
Fuente: BRUNO, JORDAN (2017).

La topología mostrada en la figura 4.9 será simulada con switches Openvswitch versión 2.1 dentro de Mininet a través de un script en lenguaje de programación escrito en Python.

La topología contiene hosts conectados a cada switch de acceso para realizar pruebas de conectividad y conexiones a los distintos hosts de la red.

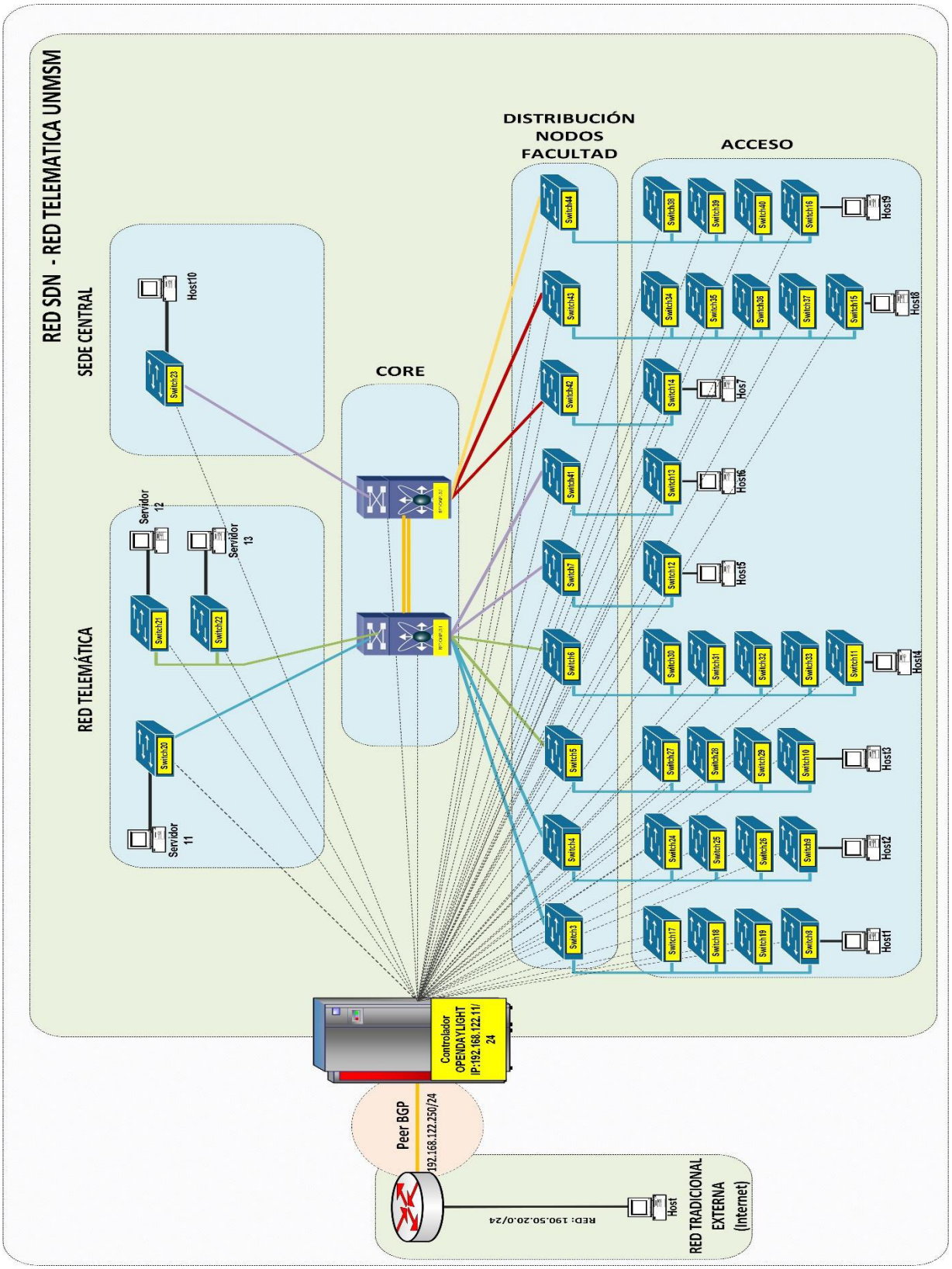

Figura 4.10: Topología de la red SDN a simular.
Fuente: Elaboración Propia

CAPITULO V

PRUEBAS DE LA TOPOLOGÍA SIMULADA Y RESULTADOS

5.1. VISUALIZACIÓN DE LA TOPOLOGÍA EN OPENDAYLIGHT

La topología propuesta se simulo en Mininet conectando los Switch OpenvSwitch v2.1 al controlador Opendaylight obteniendo la visualización por parte de la UI de la red y el reconocimiento de todos los dispositivos de la red de manera automática, el script utilizado se encuentra en el **Anexo 1**. La visualización de la topología de la red se observa en la figura 5.1:

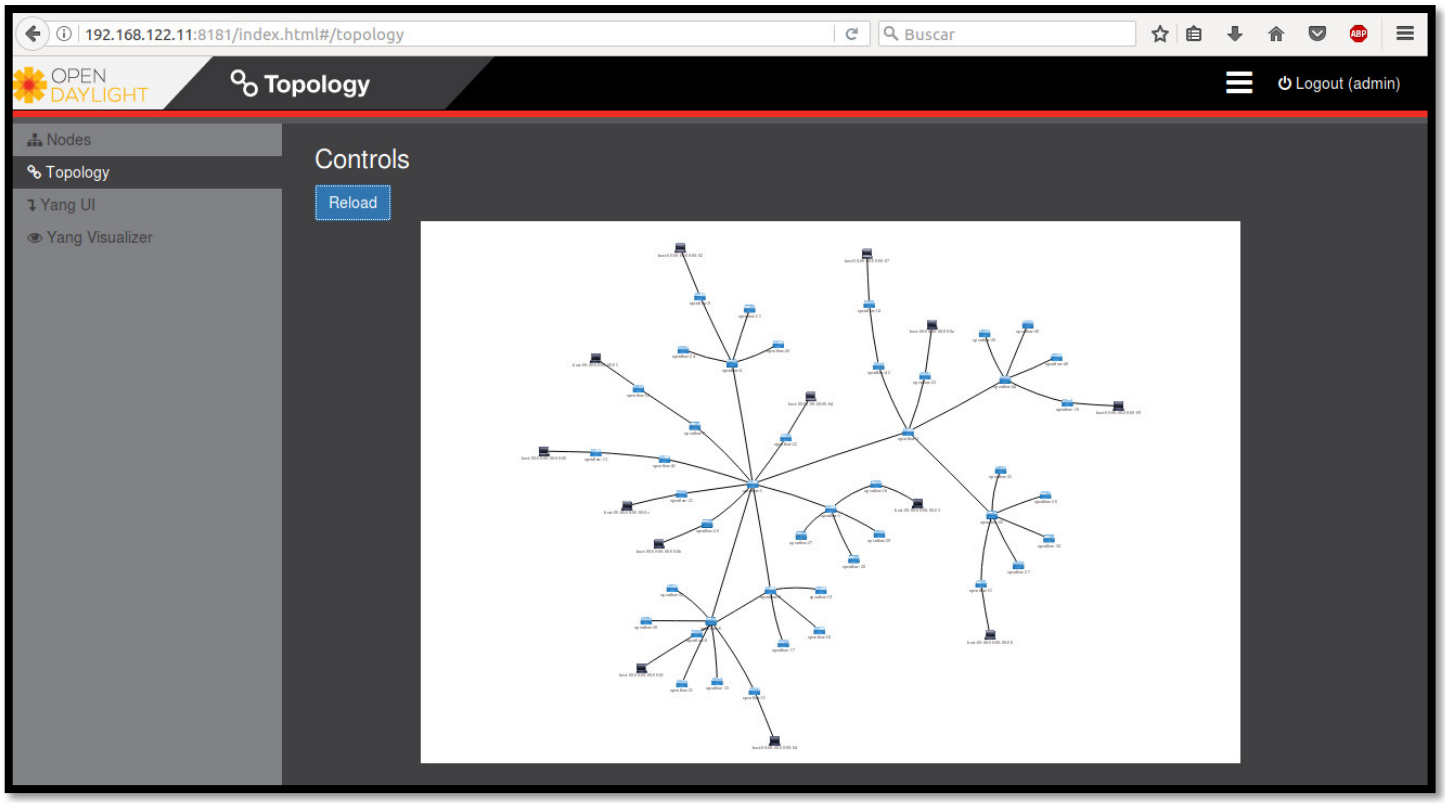

Figura 5.1: Topología de la red SDN en el módulo DLUX.
Fuente: Elaboración propia.

La topología de red mostrada en el módulo DLUX de Opendaylight muestra toda la arquitectura de la red simulada y los hosts conectados en cada switch, con lo cual provee de la información de la totalidad de la red de manera práctica y sencilla. Sin embargo, con el APP de OpenFlow Management (OFM), permite tener una mejor visualización de la red, como se muestra en la figura 5.2:

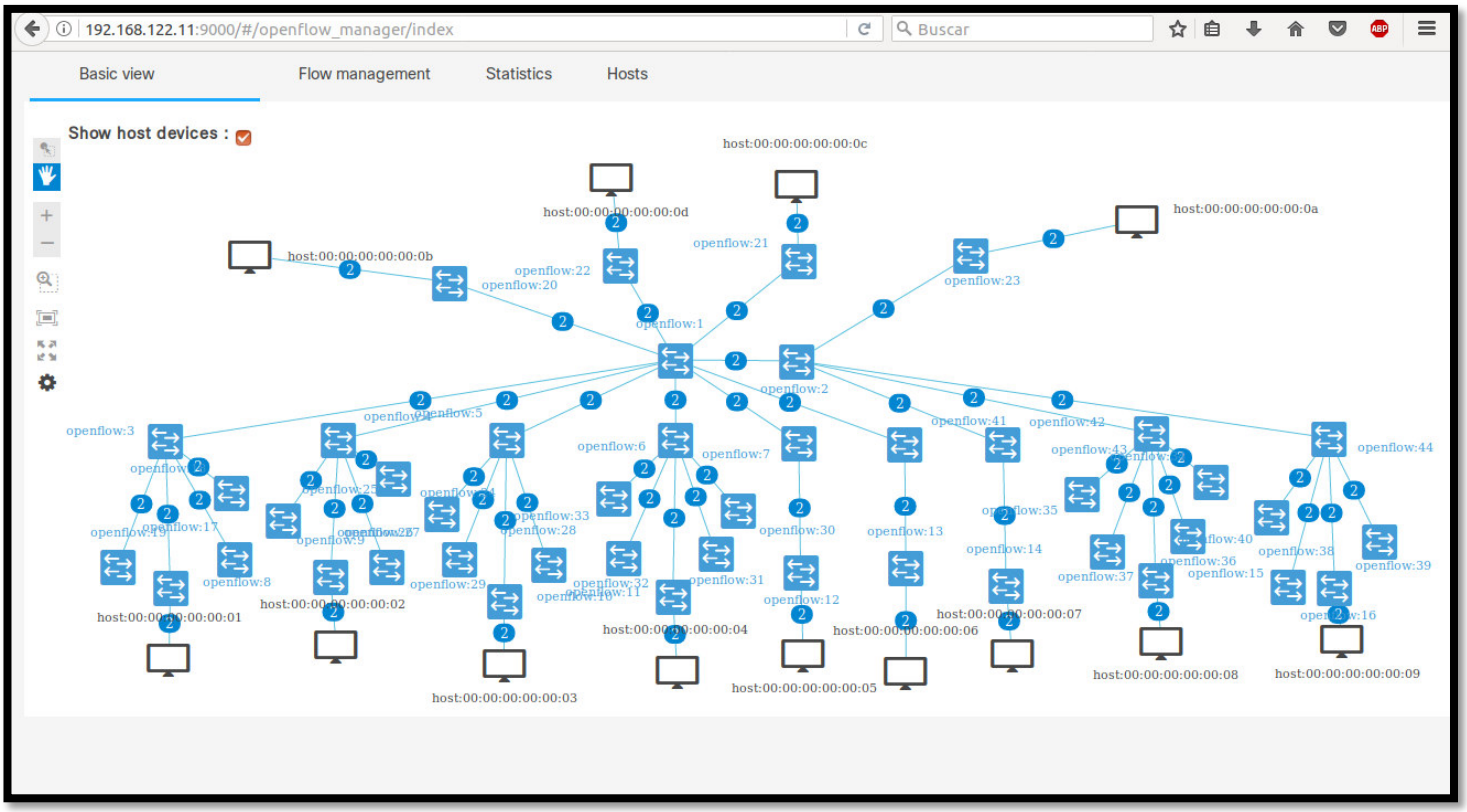

Figura 5.2: Topología de la red SDN en el AP OFM.
Fuente: Elaboración propia.

Según **MEDVED ed al. (2014)**, OpenFlow Manager (OFM) es una aplicación desarrollada para ejecutarse sobre Opendaylight para visualizar topologías Openflow, programar rutas Openflow en los dispositivos y recopilar estadísticas de Openflow. Utiliza protocolos Northbound para presentar la abstracción de la red usando REST APIs.

Las funciones secundarias de OFM, tal como se observan en la figura 5.2, son las siguientes:

- Basic View: Muestra nuestra topología SDN de los dispositivos que tienen habilitados el protocolo Openflow y se encuentran conectados al controlador, además nos permite visualizar los hosts conectados a estos dispositivos.
- Flow Management: Nos permite visualizar los flujos actuales de cada dispositivo con Openflow de la red, además de modificar, agregar o eliminar los flujos de los dispositivos.
- Statistics: Proporciona información estadística de los flujos configurados en los dispositivos, así como los puertos habilitados para Openflow.
- Hosts: Brinda información resumida de los dispositivos host habilitados para OpenFlow que administra OFM.

El APP OFM nos permitió visualizar la topología de forma más ordenada y provee de la administración total de la red desde un único punto de control ya que muestra la totalidad de los dispositivos Openflow conectados y se pudo comprobar que el controlador descubrió correctamente la topología.

Según lo comentado por **MEDVED ed al. (2014)**, uno de los objetivos clave de la arquitectura SDN en general, y habilitado en código abierto por Opendaylight, es la

abstracción de red, por lo tanto, simplifica las operaciones de red. En lugar de la administración y configuración por dispositivo, proporciona una interfaz de usuario de navegación web que los administradores de la red utilizaran para administrar la totalidad de la red a través de OpenFlow. Por tanto, el controlador SDN es el punto de control principal dentro de la red y es similar a un chasis muy escalable en el que no existe ninguna limitación en el número de ranuras físicas. La tabla 5.1 compara el controlador SDN y un dispositivo de una red tradicional:

CONTROLADOR SDN	DISPOSITIVO DE RED TRADICIONAL
Soporta cualquier tipo de hardware que tenga habilitado Openflow.	Permite la comunicación entre dispositivos a través de protocolos estándar o propietario.
Puede escalar, cantidad ilimitada de switches de acuerdo con la capacidad de hardware del controlador.	Puede escalar, sin embargo, la administración y configuración de los dispositivos se vuelve compleja.
Admite alta redundancia por varios controladores en un clúster	Admite redundancia dual o standy a través de protocolos como VRRP o propietarios.
Se comunica con los dispositivos SDN a través de protocolos southbound como Openflow, NETCONF y BGP PCEP. Y se comunica con switches router y dispositivos tradicionales fuera de SDN a través de protocolos northbound como BGP, OSPF y API directas.	Se comunica con otros routers e switches a través de protocolos estándar como BGP, OSPF o API, o propietarios limitando la compatibilidad.

Tabla 5.1: Tabla comparativa entre un controlador SDN y un dispositivo de red tradicional.
Fuente: Elaboración propia.

5.2.PRUEBAS DE CONECTIVIDAD

Mininet ofrece la posibilidad de manipular los hosts para realizar pruebas dentro de la red SDN simulada, permite ejecutar comandos tales como pin, pingall, iperf, entre otros, para realizar pruebas y analizar el comportamiento de la comunicación entre los distintos terminales y la ruta que siguen.

- **Ping:** Permite emitir paquetes ICMP entre el origen y destino con el fin de comprobar la conexión correcta de la red y comprobar la correcta conmutación de tramas de los dispositivos y del módulo L2-Switch del controlador.

- **Ping all**: Esta prueba permite realizar ping desde un terminal hacia todos los de la red, de esta forma se identifica a conexión total en la red y su visualización de todos los terminales.

- **Iperf:** Es una herramienta multiplataforma que puede producir mediciones de rendimiento estandarizadas para cualquier red. Iperf tiene funcionalidad de cliente

y servidor, puede crear flujos de datos para medir el rendimiento entre los dos extremos en una o ambas direcciones.

5.2.1. PRUEBAS ICMP Y FLUJOS OPENFLOW

Para el análisis de la topología SDN se tomaron los tiempos de respuesta de cada host hacia los host servidores. En esta prueba, las IPs y MACs de los hosts fueron los presentados en la tabla 5.2.

NOMBRE DEL HOST (NODO)	IP/MASK	MAC
h1	10.0.0.1/8	00:00:00:00:00:01
h2	10.0.0.2/8	00:00:00:00:00:02
h3	10.0.0.3/8	00:00:00:00:00:03
h4	10.0.0.4/8	00:00:00:00:00:04
h5	10.0.0.5/8	00:00:00:00:00:05
h6	10.0.0.6/8	00:00:00:00:00:06
h7	10.0.0.7/8	00:00:00:00:00:07
h8	10.0.0.8/8	00:00:00:00:00:08
h9	10.0.0.9/8	00:00:00:00:00:09
h10 (Sede Central)	10.0.0.10/8	00:00:00:00:00:0A
h11 (Servidor)	10.0.0.11/8	00:00:00:00:00:0B
h12 (Servidor)	10.0.0.12/8	00:00:00:00:00:0C
h13 (Servidor)	10.0.0.13/8	00:00:00:00:00:0D

Tabla 5.2: Direcciones IP y MAC de los hosts simulados.
Fuente: Elaboración propia.

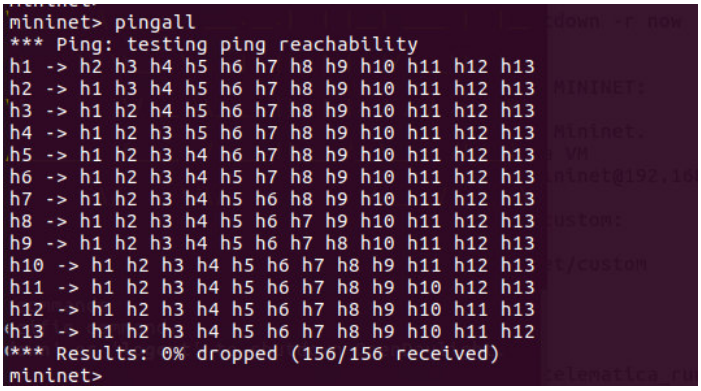

Figura 5.3: Pruebas de Ping-All exitosas.
Fuente: Elaboración propia.

Las pruebas de conectividad se realizaron mediante el comando ping, los resultados fueron los siguientes (ver figura 5.4):

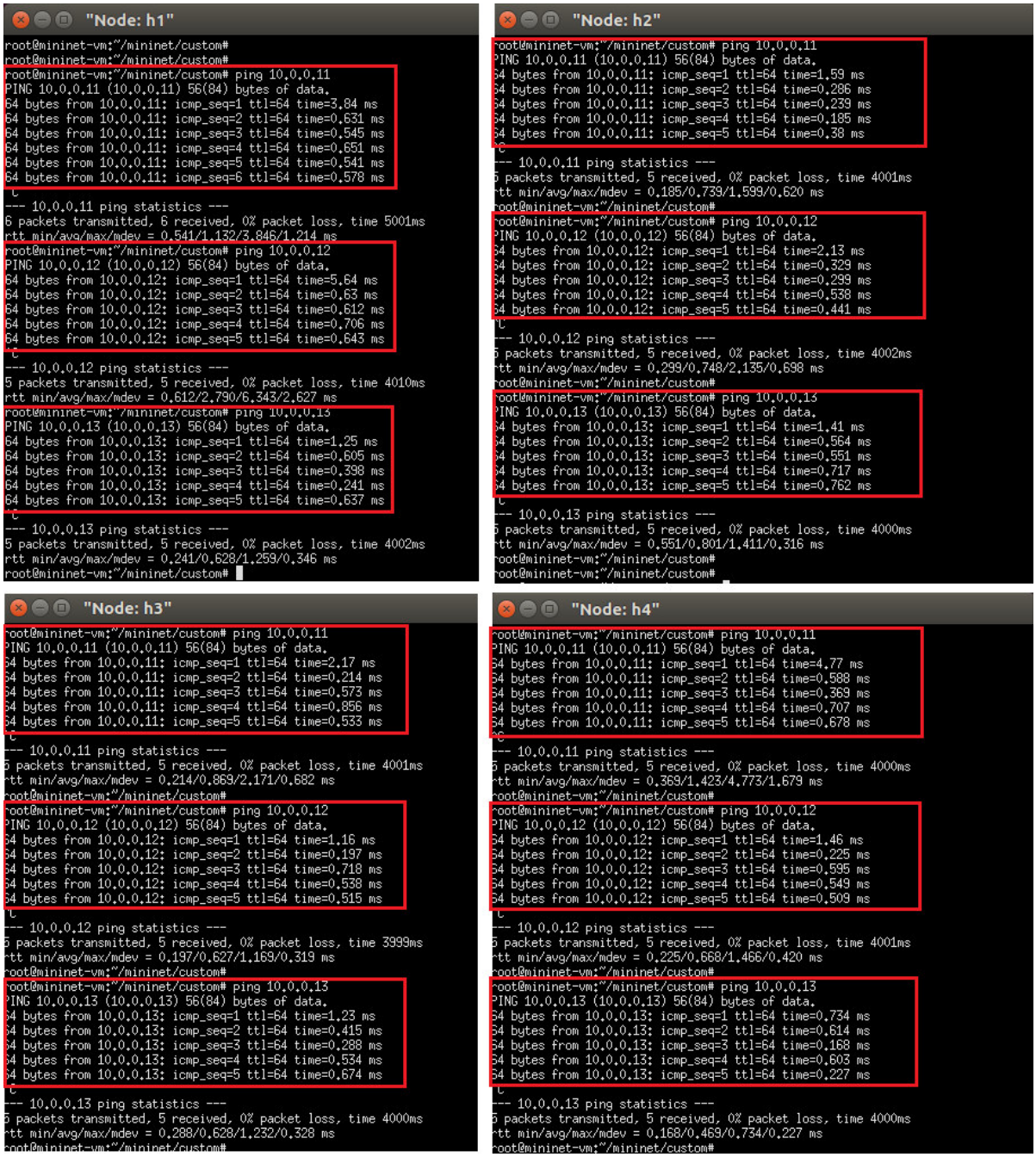

Figura 5.4: Pruebas de ICMP de los hosts 1,2,3 y 4 hacia los servidores.
Fuente: Elaboración propia.

La prueba de conectividad ICMP se replicaron en todos los hosts de la topología y se recopilo la información en las tablas 5.3; 5.4 y 5.5 para los tiempos de respuesta hacia los servidores h11, h12 y h13 respectivamente:

Nombre del Host (Nodo)	TIEMPO DE RESPUESTA ICMP HACIA EL HOST h11 (mseg)				
	1ra	2da	3ra	4ta	5ta
h1	4.62	0.527	0.245	0.252	0.425
h2	2.34	0.254	0.337	0.752	0.675
h3	1.85	0.239	0.423	0.248	0.253
h4	5.79	0.348	0.458	0.582	0.752
h5	3.24	0.625	0.255	0.325	0.485
h6	2.43	0.52	0.654	0.851	0.425
h7	4.41	0.582	0.808	0.678	0.685
h8	2.51	0.342	0.249	0.485	0.145
h9	3.72	0.475	0.385	0.245	1.01
h10	6.34	0.312	0.256	0.981	0.125
PROMEDIO	**3.725**	**0.4224**	**0.407**	**0.5399**	**0.498**

Tabla 5.3: Tiempo de respuesta (mseg) hacia el host h11.
Fuente: Elaboración propia.

Nombre del Host (Nodo)	TIEMPO DE RESPUESTA ICMP HACIA EL HOST h12 (mseg)				
	1ra	2da	3ra	4ta	5ta
h1	6.25	0.453	0.569	0.747	0.227
h2	3.24	0.375	0.466	0.214	0.679
h3	2.25	0.211	0.678	0.588	0.763
h4	2.48	0.217	0.523	0.545	0.223
h5	3.12	0.385	0.652	0.611	0.539
h6	1.05	0.436	0.486	0.148	0.168
h7	2.71	0.478	0.469	0.105	0.643
h8	3.94	0.493	0.851	0.209	0.551
h9	1.07	0.598	0.125	0.094	0.244
h10	4.73	0.725	0.149	0.248	0.781
PROMEDIO	**3.084**	**0.4371**	**0.4968**	**0.3509**	**0.4818**

Tabla 5.4: Tiempo de respuesta (mseg) hacia el host h12.
Fuente: Elaboración propia.

Nombre del Host (Nodo)	TIEMPO DE RESPUESTA ICMP HACIA EL HOST h13 (mseg)				
	1ra	2da	3ra	4ta	5ta
h1	2.78	0.505	0.199	0.301	0.154
h2	2.411	0.378	0.233	0.555	0.521
h3	2.832	0.481	0.269	0.231	0.29
h4	1.714	0.345	0.104	0.64	0.395
h5	2.331	0.781	0.519	0.56	0.264
h6	1.781	0.284	0.639	0.617	0.198
h7	3.712	0.158	0.278	0.502	0.406
h8	1.791	0.539	0.438	0.488	0.367
h9	0.851	0.254	0.471	0.547	0.646
h10	0.912	0.871	0.19	0.498	0.151
PROMEDIO	**2.1115**	**0.4596**	**0.334**	**0.4939**	**0.3392**

Tabla 5.5: Tiempo de respuesta (mseg) hacia el host h12.
Fuente: Elaboración propia.

Según lo observado en las tablas 5.3; 5.4 y 5.5, el primer paquete enviado tiene un tiempo de respuesta más elevado comparado con los subsiguientes paquetes enviados (mayor a 1ms) en todos los casos. El primer paquete enviado tiene un tiempo de respuesta mayor a 1ms porque el Switch Openflow aún no tiene las tablas de flujos necesarias para conmutar el paquete.

El primer paquete enviado por el host es un mensaje ARP que es encapsulado por el switch de acceso y enviado al controlador Openflow a través de un mensaje "Packet_in". El paquete ARP solo se propaga hasta el switch de acceso.

El controlador Opendaylight recibe el paquete y utiliza el módulo L2-Switch para procesar la solicitud. El controlador envía un mensaje "Packet_out" para enviar un mensaje broadcast por todos los puertos del switch menos por el puerto por el cual recibió el paquete. La respuesta ARP se reenvía al host origen, que fue el ARP principal consultado.

Con la información de la trama completa, el host origen envía los paquetes ICMP con las direcciones de capa 3 y capa 2 del destino. El switch Openflow encapsulara nuevamente este paquete para enviarlo al controlador Openflow a través del mensaje "Packet_In". El controlador Opendaylight recibe el paquete y realiza la búsqueda en la base de datos "Address_Tracker" (figura 5.4) del módulo L2-Switch para ubicar al host destino dentro de la topología. Una vez que el switch Openflow recibe el mensaje "Packet_Out", desencapsula el paquete original y lo procesa, renviándolo hacia el destino. De este modo, las tablas de flujo quedan instalas en el switch Openflow y los siguientes paquetes ICMP enviados tienen un tiempo de respuesta menor a un 1ms.

El módulo L2-Switch actualizara las tablas de flujos de acuerdo los eventos de la red, por ejemplo, ante la caída de un enlace o de un puerto o la inclusión de un nuevo host en la

red. Ver el **Anexo 2** para ver las tablas de flujo del Switch 22 en el controlador Opendaylight.

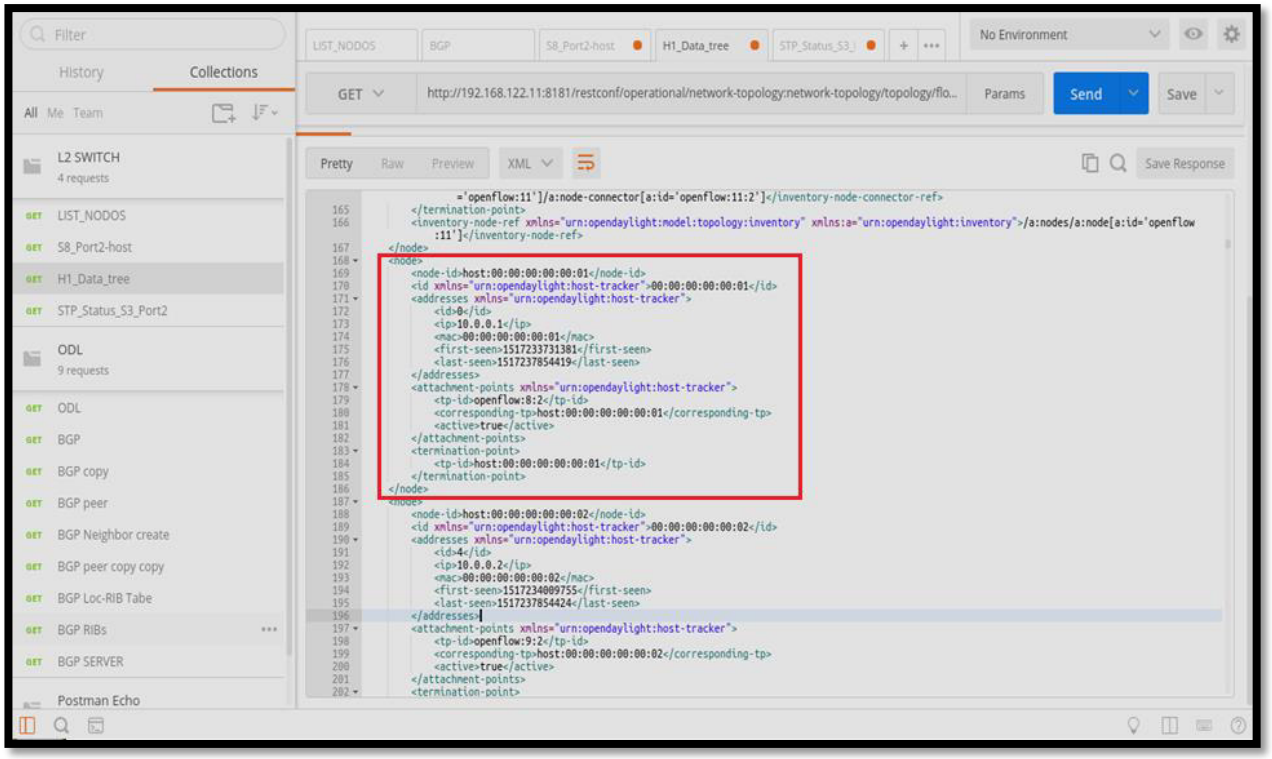

Figura 5.5: Base de datos “Address-tracker” del módulo L2-Switch.
Fuente: Elaboración propia.

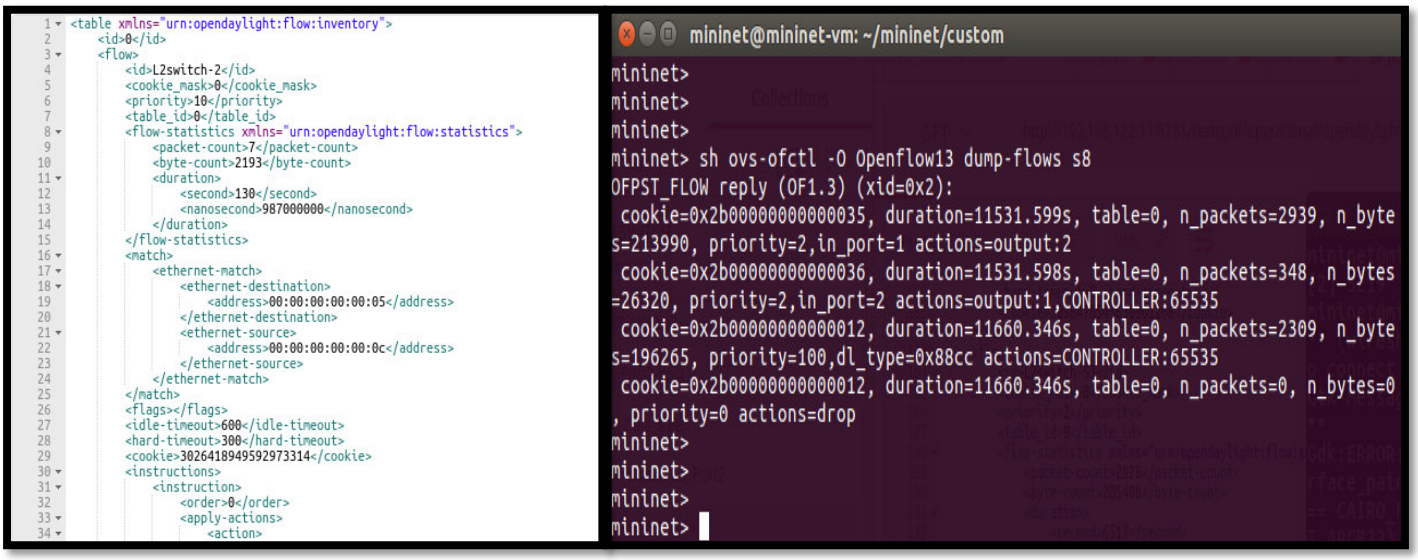

Figura 5.6: Entradas de flujo en el controlador SDN y switch SDN.
Fuente: Elaboración propia.

Se pudo comprobar que el controlador Opendaylight instalo correctamente las entradas de flujos en los switch Openflow de forma automática y sin intervención a nivel local del dispositivo de red.

5.2.2. PRUEBAS ICMP ENTRE SUB-REDES

En esta prueba se realizó el cambio de las direcciones IP del host “h1” y del host red telemática “h12” a la subred 192.168.1.0/24 con las direcciones IP de la tabla 5.6, y se realizaron las pruebas de conectividad con el host4 de la subred 10.0.0.0/8 mediante el comando ping.

Nombre del Host (Nodo)	IP	MAC
h1	192.168.1.10/24	00:00:00:00:00:01
h4	10.0.0.4/8	00:00:00:00:00:04
h12 (Red Telemática)	192.168.1.120/24	00:00:00:00:00:0C

Tabla 5.6: Direcciones IPs y MACs de los host h1, h4 y h12
Fuente: Elaboración propia.

El módulo L2-Switch instalo las entradas de flujos necesarios para la comunicación entre los hosts y se comprobó que la conectividad es posible (ver figura 5.7) y sin perdidas de paquetes.

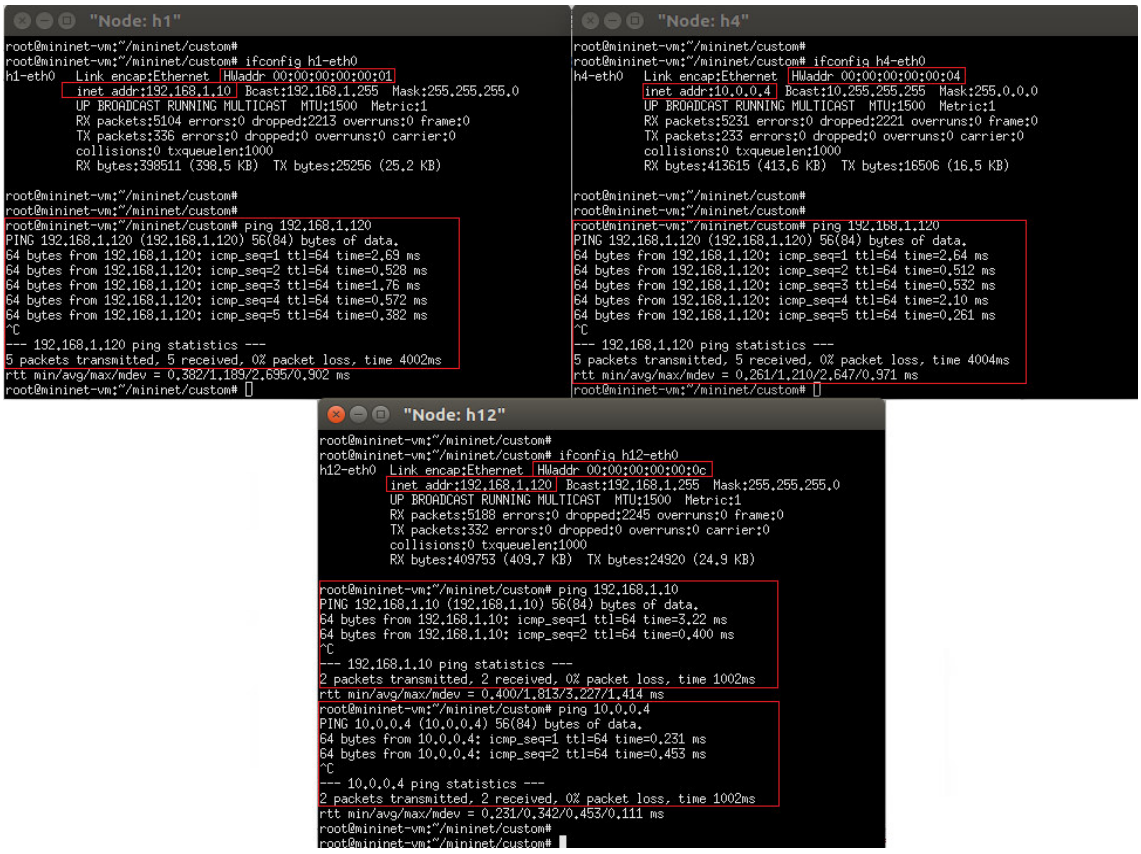

Figura 5.7: Conectividad entre host de distintas sub-redes.
Fuente: Elaboración propia.

5.2.3. PRUEBAS DE TRAFICO TCP (BW) Y UDP (JITTER)

Se realizaron pruebas de tráfico TCP y UDP entre un host y el servidor para medir el rendimiento en términos de ancho de banda y jitter. Se utilizó la herramienta IPERF, utilizada ampliamente para medir y ajustar el rendimiento de una red, puede crear flujos de datos para medir el rendimiento entre un host y un servidor en forma unidireccional o bidireccional. Los resultados típicamente contienen el ancho de banda utilizado, la transferencia de datos, el tiempo utilizado, el jitter, entre otros valores dependiendo de la prueba ejecutada.

5.2.3.1. TRAFICO TCP (BW)

Se realizaron las pruebas IPERF utilizando el protocolo TCP de capa 4 del modelo OSI, con un tamaño de ventana de 85.3 Kbyte. Se configuro el host “h12” como servidor utilizando el puerto TCP8080, y el host “h1” y “h10” como clientes para enviar datos al puerto TCP8080 por 20 segundos. Los resultados fueron los mostrados en las figuras 5.8 y 5.9.

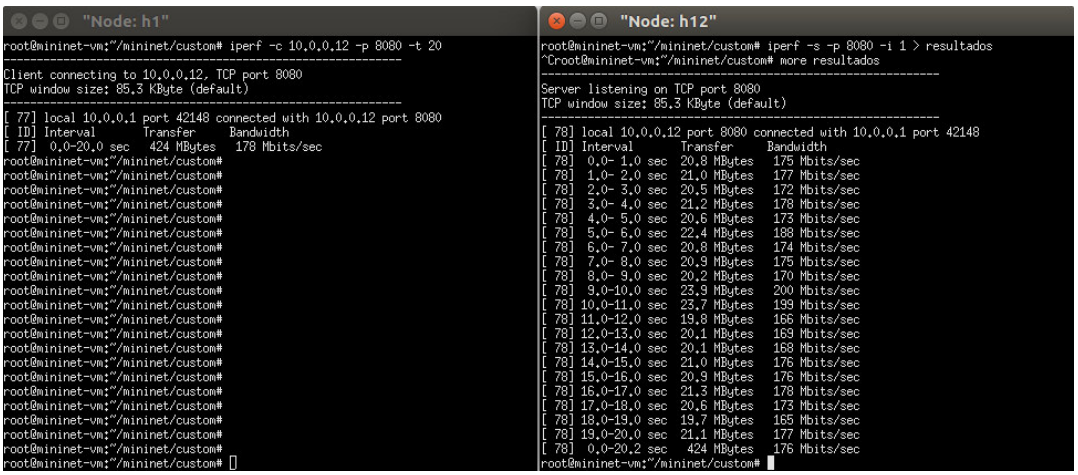

Figura 5.8: Pruebas IPERF TCP host cliente “h1” host red telemática “h12”.
Fuente: Elaboración propia.

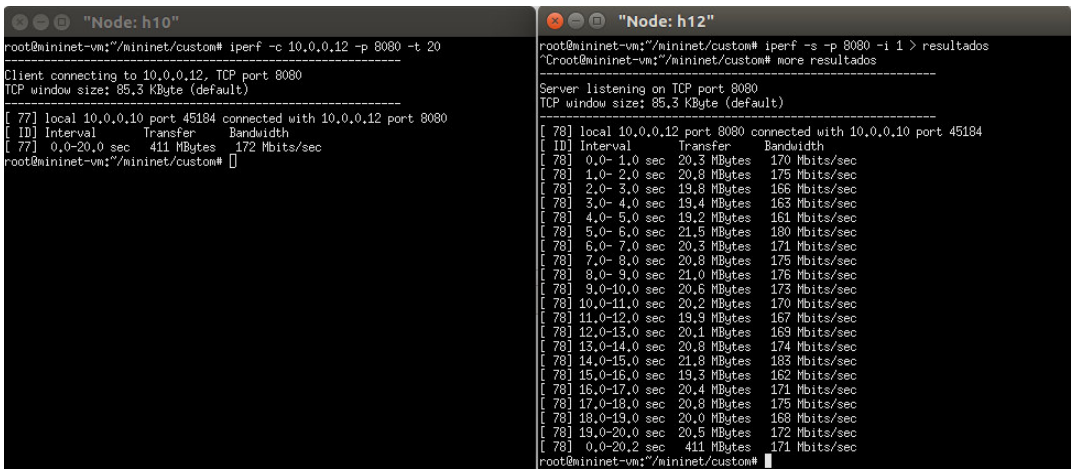

Figura 5.9: Pruebas IPERF TCP host cliente “h1” host red telemática “h12”.
Fuente: Elaboración propia.

Se graficaron los resultados utilizando Gnuplot, los resultados fueron los siguientes:

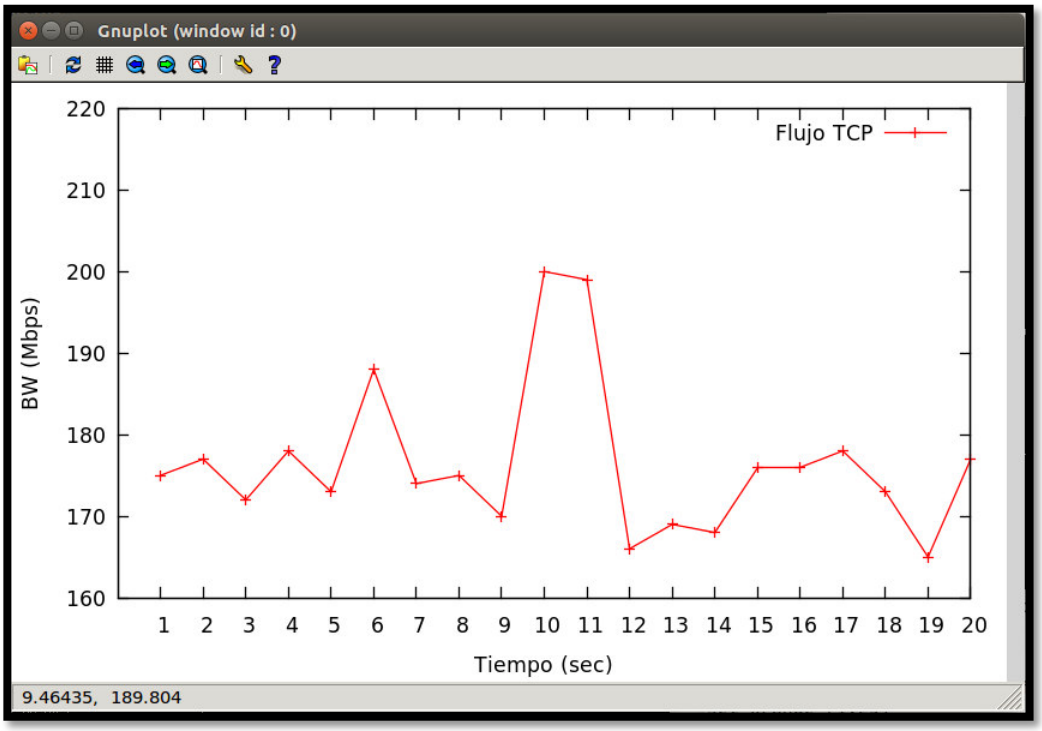

Figura 5.10: Grafica de BW entre “h1” y “h12”.
Fuente: Elaboración propia.

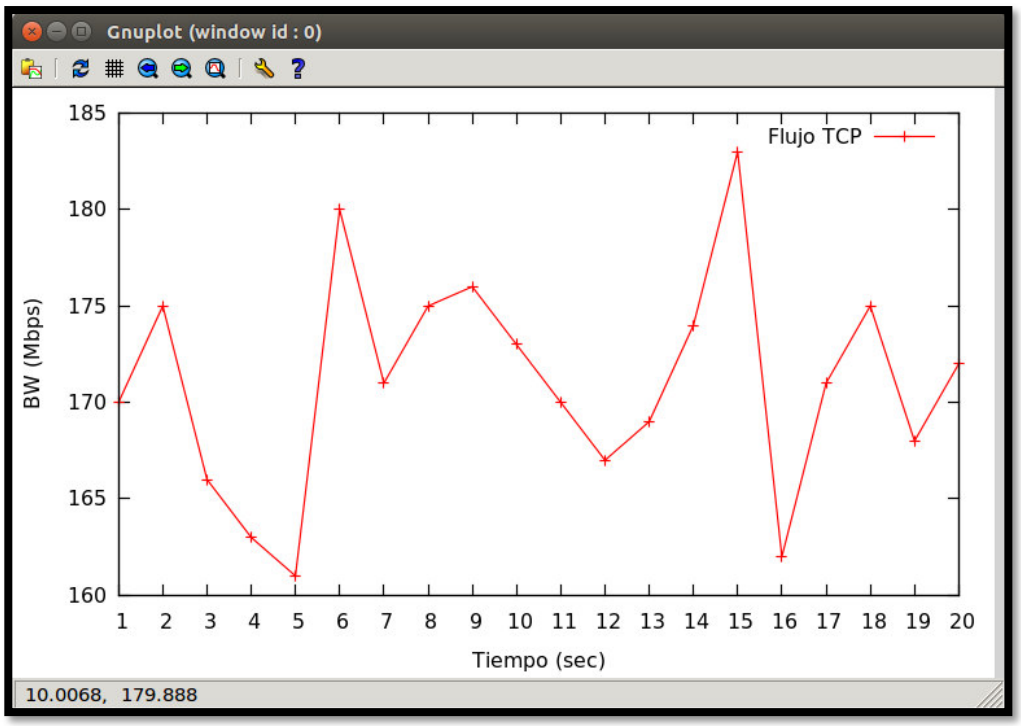

Figura 5.11: Grafica de BW entre “h10” y “h12”.
Fuente: Elaboración propia.

De las figuras 5.10 y 5.11 se pueden observar que, entre los hosts “h1” y “h10” (red campus) y el servidor “h12” (Red Telemática), se tienen un BW promedio entre 160 Mbps y 200 Mbps, por lo cual se puede manifestar que la red SDN simulada puede manejar tráfico TCP a un BW de 160 Mbps aproximadamente como mínimo. El protocolo de transmisión TCP es usado por varios servicios de red básicos, tales como, web, correo, ftp, etc.

5.2.3.2. TRAFICO UDP (JITTER)

Se realizaron las pruebas IPERF utilizando el protocolo UDP de capa 4 del modelo OSI, con un tamaño de buffer de 208 KBytes. Se configuro el host “h12” como servidor utilizando el puerto UDP8081, y el host “h1” y “h10” como clientes para enviar datos al puerto UDP8081 por 20 segundos.

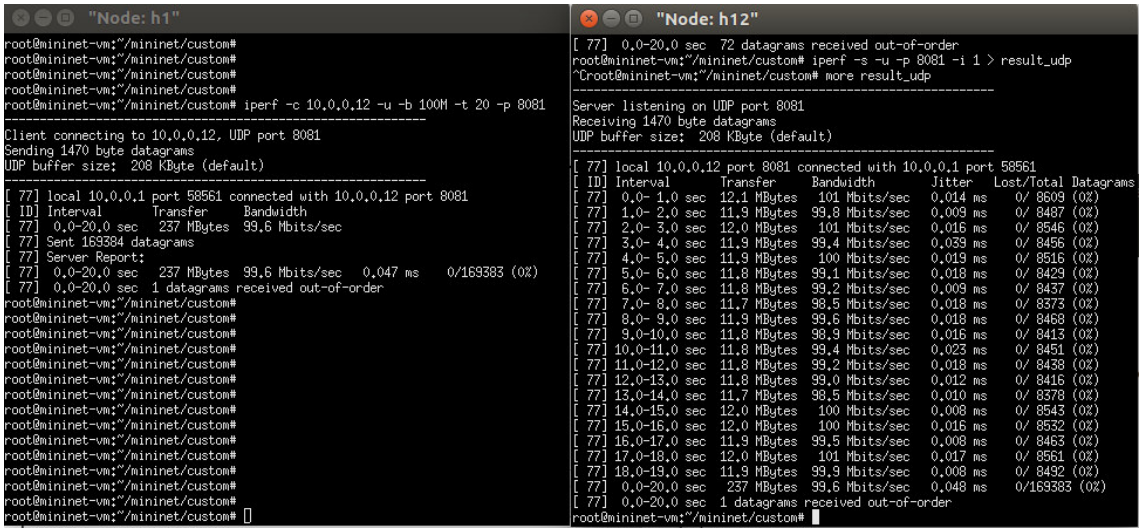

Figura 5.12: Pruebas IPERF UDP host cliente “h1” host “h12”.
Fuente: Elaboración propia.

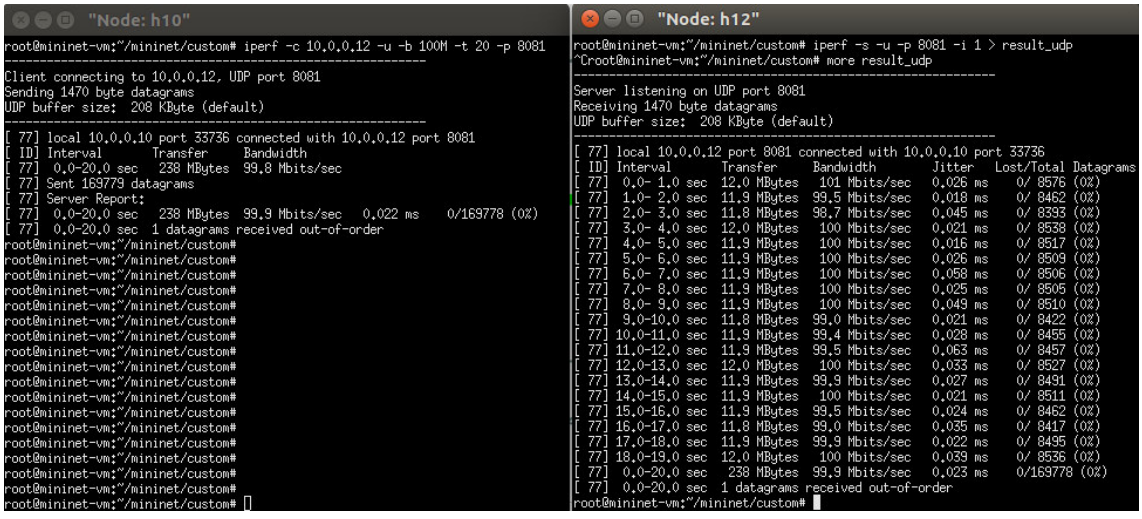

Figura 5.13: Pruebas IPERF UDP host cliente “h1” host “h12”.
Fuente: Elaboración propia.

Los resultados de la simulación, utilizando Gnuplot, son los que se muestran en las Figuras 5.14 y 5.15.

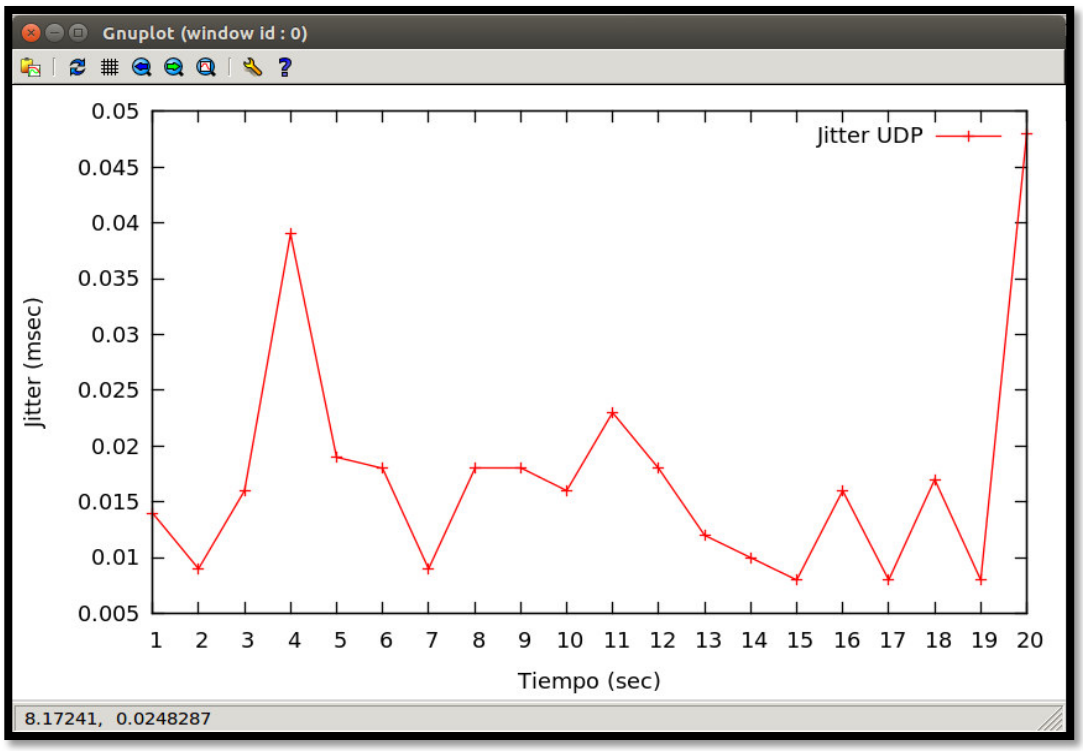

Figura 5.14: Pruebas IPERF UDP host cliente "h1" host "h12".
Fuente: Elaboración propia.

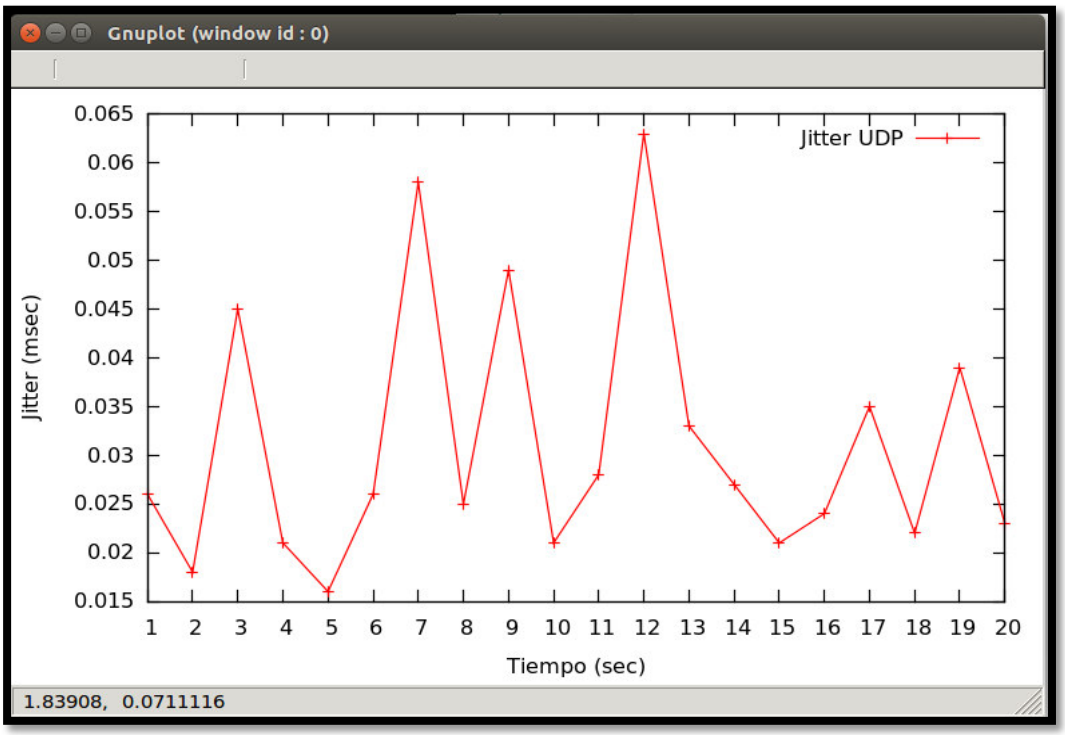

Figura 5.15: Pruebas IPERF UDP host cliente "h10" host "h12".
Fuente: Elaboración propia.

En estas figuras 5.14 y 5.15 se puede observar que, entre los hosts “h1” y “h10” (red campus) y el servidor “h12” (Red Telemática), se tienen un BW promedio entre 160 Mbps y 200 Mbps, por lo cual se puede manifestar que la red SDN simulada puede manejar tráfico UDP con un jitter menor a 0.065 msec. El protocolo de transmisión UDP es usado para las transmisiones de voz sobre IP (VoIP), entre otros.

5.2.3.3. TRAFICO WEB

Se configuro un servidor WEB en el servidor “h12” y se realizaron conexiones HTTP desde los hosts de la Red Campus. La respuesta HTTP desde el host “h5” fue correcta (ver figura 5.16).

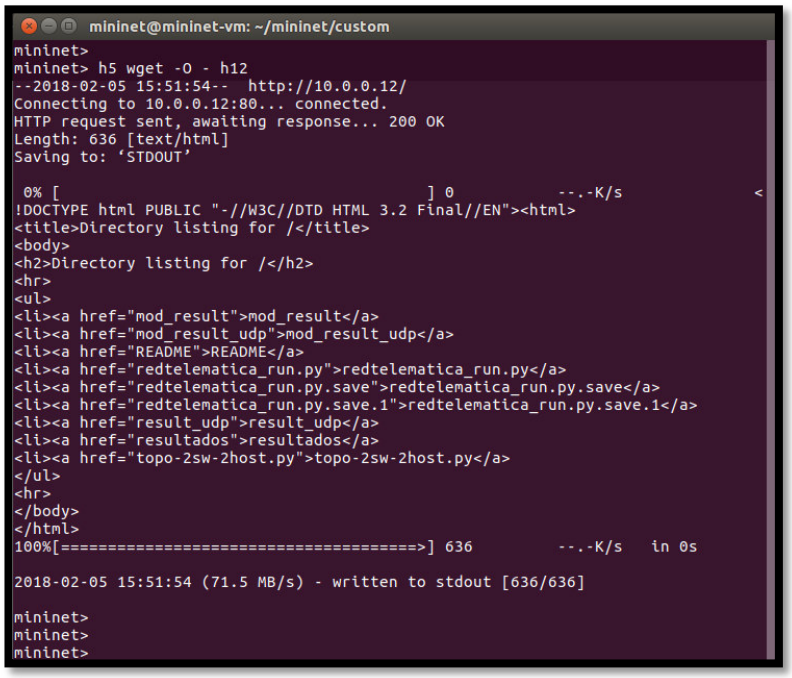

Figura 5.16: Conexión HTTP del host “h5” al servidor “h12”.
Fuente: Elaboración propia.

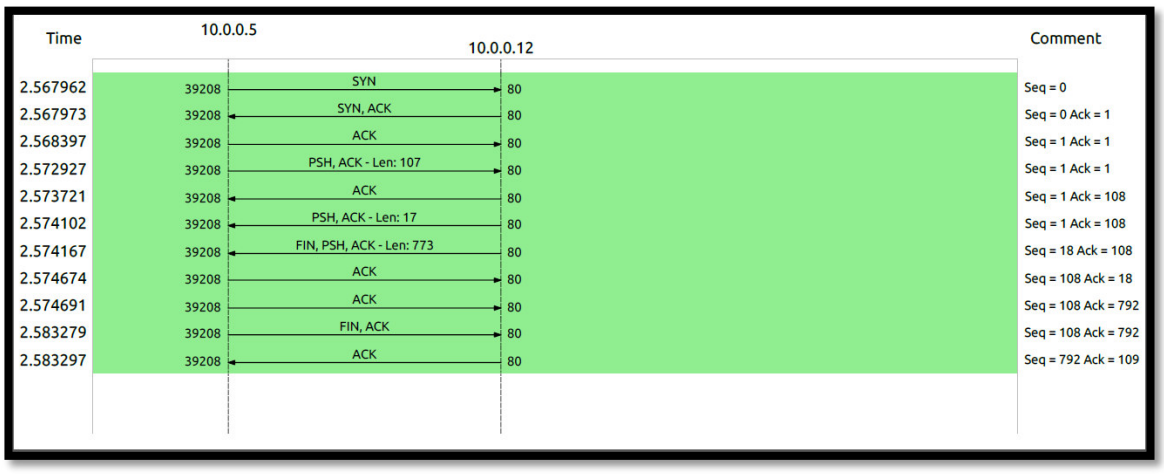

Figura 5.17: Hand-shake y flujo de la conexión HTTP visualizado en Wireshark.
Fuente: Elaboración propia.

De acuerdo con la figura 5.17, la conexión HTTP sobre la red SDN se realizó sin inconvenientes.

5.3. CONECTIVIDAD CON REDES TRADICIONALES – BGP PCEP

Se implementó el protocolo de enrutamiento BGP con el soporte del módulo BGP-PCEP instalado en el controlador Opendaylight. El protocolo BGP implementado incluye mensajes y los atributos de las RFC4271, RFC4760, RFC1997 y RFC4360.

En la figura 5.18 muestra el diagrama entre los módulos BGP en Opendaylight, La implementación de BGP dentro del controlador Opendaylight consiste en múltiples módulos y características que se comunican entre sí para proporcionar la compatibilidad con el protocolo de enrutamiento.

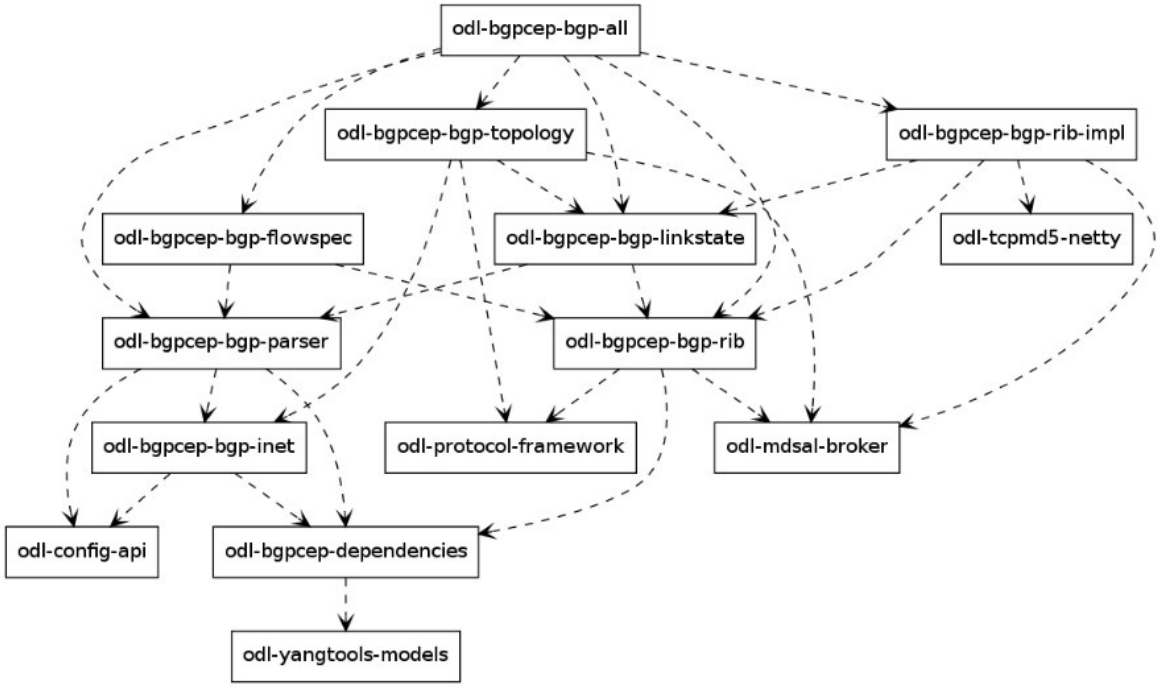

Figura 5.18: Módulos y características de BGP-PCEP.
Fuente: OPENDAYLIGHT (2017).

El protocolo BGP intercambia prefijos de red (rutas) entre pares BGP. Opendaylight almacena la información de los prefijos de red en la "Routing Information Base" (RIB) y la visualización general de la topología de la red. Según las políticas configuradas, selecciona las rutas potenciales dentro de la RIB.

Para las pruebas, se simulo un Router tradicional con el sistema operativo VyOS v1.1.8. Según **VYOS (2018),** VyOS es un sistema operativo de red de código abierto, basado en GNU / Linux siendo muy similar a los Reuters tradicionales de hardware, focalizado en un buen soporte para características avanzadas de enrutamiento como protocolos de enrutamiento dinámico e interfaz de línea de comandos (CLI).

La topología implementada se muestra en la figura 5.19:

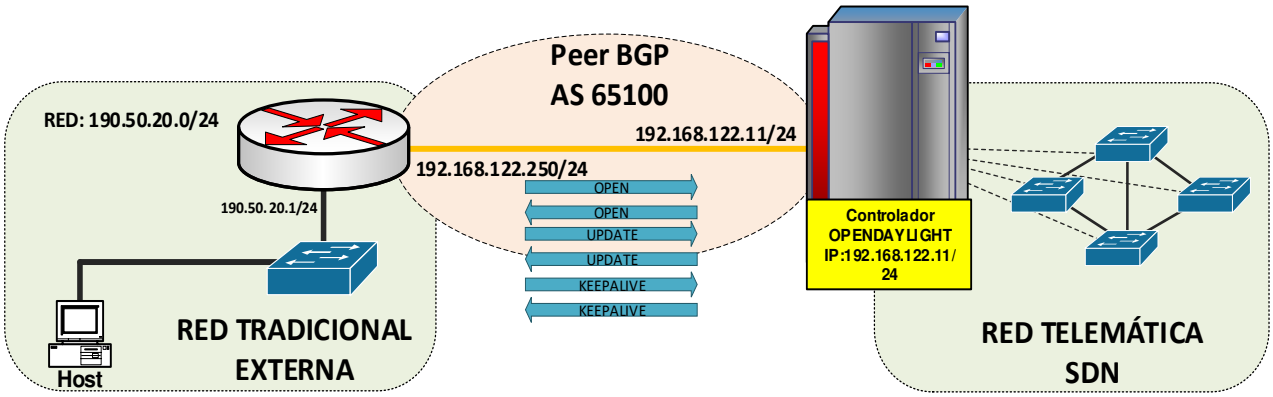

Figura 5.19: Topología de simulación SDN.

Fuente: Elaboración propia.

La configuración realizada del módulo BGP-PCEP se encuentra en el **Anexo 3,** se utilizó el número de sistema autónomo AS 65100 para el intercambio de prefijos de red y se configuro los peer BGP en el controlador y en el Router VyOS. VyOS envió el prefijo de red 190.50.20.0/24 al controlador Opendaylight, el prefijo de red se visualizó en la tabla RIB del controlador (ver figura 5.20).

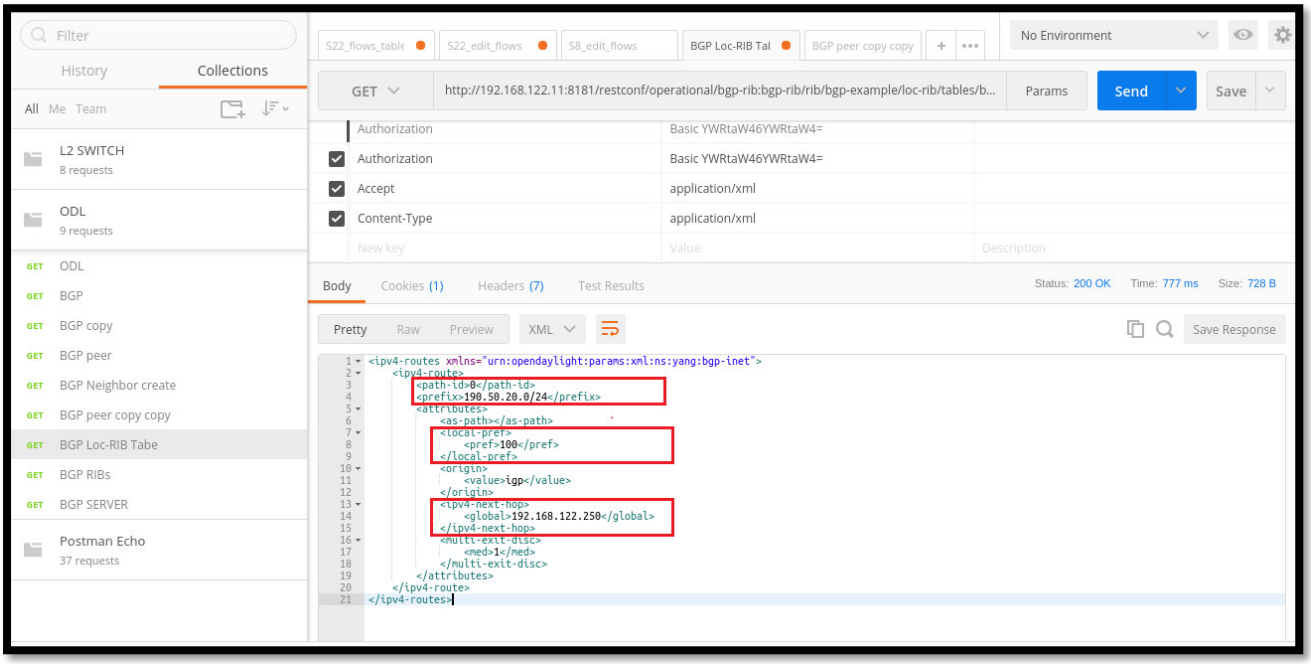

Figura 5.20: BGP Tabla local RIB en Opendaylight.
Fuente: Elaboración propia.

El peer BGP del VyOS - 192.168.122.250/24 se instaló en el módulo BGP-PCEP, se visualizó su correcto funcionamiento en el controlador (ver figura 5.21).

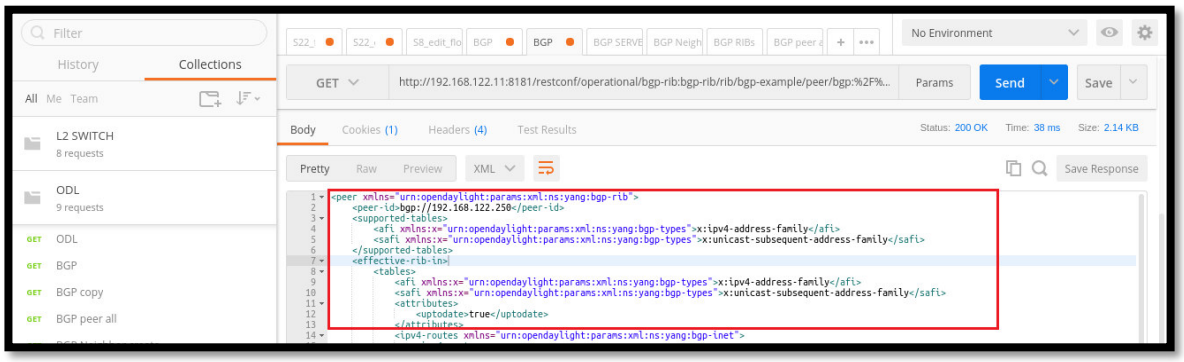

Figura 5.21: Peer BGP activo en Opendaylight.
Fuente: Elaboración propia.

Se comprobó que el Router VyOS estableció una correcta sesión BGP, el Router VyOS envió los prefijos de red 192.168.50.20/24 y recibió el prefijo de red 10.0.0.11/32 del controlador Opendaylight a través de mensajes Update (ver figura 5.22).

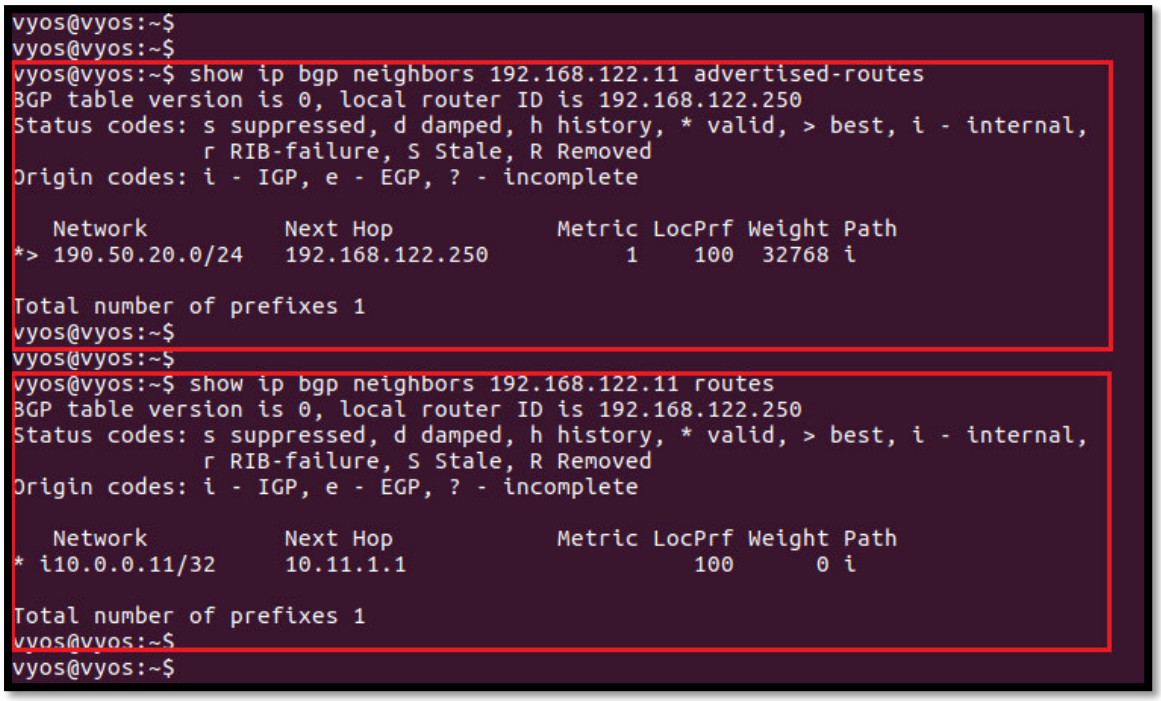

Figura 5.22: Router VyOS, sesión BGP por CLI.
Fuente: Elaboración propia.

La implantación de BGP permitió a la red SDN comunicarse con dispositivos de red tradicionales, como routers. El controlador Opendaylight permitió la administración y gestión de protocolos de enrutamiento tradicionales para interconectar el dominio SDN con los dominios de red tradicionales, las rutas de red intercambiadas a través del protocolo BGP se pueden administrar desde el mismo controlador sin intervención de otro dispositivo de red.

5.4.INDICADORES DE LA MEJORA DE LA GESTIÓN

La red telemática comprende una gran cantidad de switches, routers, firewalls y numerosos tipos de características con diferentes eventos que ocurren simultáneamente. El administrador de red es responsable de configurar la red para hacer cumplir varias políticas de alto nivel y responder a la amplia gama de eventos de red (por ejemplo, cambios de tráfico, intrusiones, calidad de servicio) que pueden ocurrir.

La configuración de una red con arquitectura tradicional es compleja porque la implementación de estas políticas de alto nivel requiere que se especifiquen en términos de configuración distribuida de bajo nivel. Las redes tradicionales ofrecen un limitado rango de mecanismos para responder automáticamente a la amplia gama de eventos que pueden ocurrir dificultando la gestión de los dispositivos. Los administradores deben implementar políticas y tareas complejas cada vez más sofisticadas con un conjunto limitado y altamente restringido de comandos de configuración de dispositivos de bajo nivel en un entorno de interfaz de línea de comandos (CLI). Sin embargo, en muchos casos el estado de la red cambia continuamente y los administradores de red deben ajustar manualmente la configuración de la red en respuesta a las condiciones cambiantes de la red. Debido a esta limitación, los administradores de red utilizan herramientas externas o incluso crean scripts ad hoc para reconfigurar dinámicamente los dispositivos de red cuando ocurren eventos. Según **HYOJOON, NICK (2013)**, los cambios de configuración son frecuentes y difíciles de manejar, lo que lleva a configuraciones erróneas.

Para nuestra comparación, se realizó una simulación de la red telemática con una arquitectura de red tradicional en el simulador GNS3 usando switches del proveedor Cisco. Los equipos usados fueron los siguientes:

NOMBRE	PROVEEDOR	VERSIÓN DE IOS
Cisco NX-OS	CISCO	v 7.3.0
Cisco L2 IOSv	CISCO	v 15.6(1)T

Tabla 5.7: Versión de switch utilizados para la simulación.
Fuente: Elaboración propia.

Se habilitaron las siguientes características básicas para la conectividad y seguridad entre los dispositivos de red:

- Switchport
- 802.1q trunk, 802.1q vlans
- Spanning Tree
- Port-Channel
- Port-ACLs
- DHCP Snooping
- IP device tracking
- Layer-3 forwarding

- Routing protocol support
- Inter-vlan routin

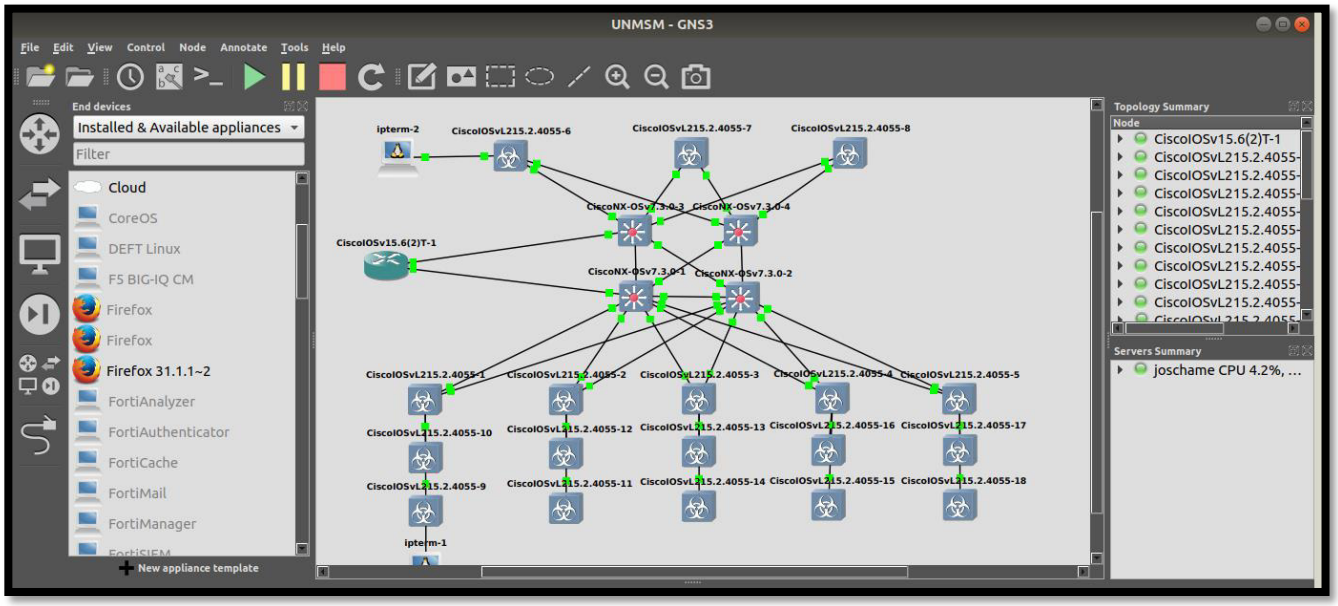

Figura 5.23: Simulación de la red telemática con una arquitectura de red tradicional.
Fuente: Elaboración propia.

Luego de las configuraciones se verifico la conectividad entre los hosts con éxito, así como la administración de cada dispositivo de red

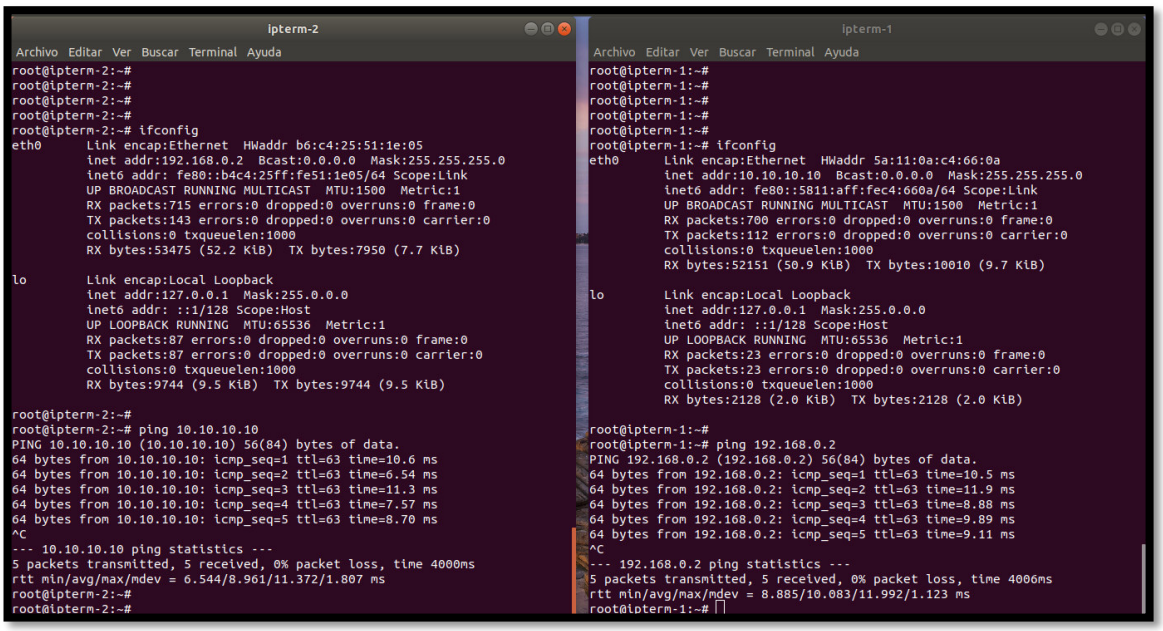

Figura 5.24: Pruebas de conectividad de la simulación de la red tradicional.
Fuente: Elaboración propia.

5.4.1. TIEMPOS DE IMPLEMENTACIÓN POR DISPOSITIVO

A continuación, se describe los tiempos de implementación aproximadas en las simulaciones presentadas:

- Simulación de la red de arquitectura tradicional

El tiempo de implementación de cada equipo se llevó a cabo en 5 a 7 minutos para una configuración básica, sin embargo, si se realiza alguna configuración más avanzada como mapeo por listas de acceso, protocolos de enrutamiento, etc., la configuración tomaría más tiempo complicando la gestión de cada equipo.

- Simulación de la red de arquitectura SDN

En la red SDN simulada, los switches Openflow fueron configurados con la IP del controlador y obtuvieron sus tablas de flujos desde el controlador por el módulo L2-SWITCH. Según las pruebas, las tablas de flujo se instalaron en aproximadamente 5 mseg para proveer la conectividad a los hosts.

ARQUITECTURA DE RED	TIEMPO DE IMPLEMENTACIÓN POR DISPOSITIVO
TRADICIONAL	5 - 7 minutos aprox.
SDN	5 mseg aprox.

Tabla 5.8: Tiempo de implementación por dispositivo.
Fuente: Elaboración propia.

Según la tabla 5.8, debido a que el plano de control de los dispositivos se encuentra unificado, los tiempos de implementación de un dispositivo SDN es menor que el tiempo de implementación de un dispositivo de red tradicional. La reducción del tiempo mejora la gestión de la red automatizando las tareas de implementación de nuevos dispositivos a la red.

5.4.2. TIEMPOS DE DESPLIEGUE DE CONFIGURACIÓN

A continuación, se describe los tiempos aproximados de configuración de nuevas políticas de red en las simulaciones presentadas:

- Simulación de la red de arquitectura tradicional

En la red tradicional simulada se realizó el despliegue de configuración del protocolo BGP, listas de acceso para bloqueo de puertos específicos y una nueva VLAN en los dispositivos de la red. La configuración fue realizada mediante línea de comandos (CLI) de cada dispositivo de red tomando un tiempo aproximado de 5 minutos por dispositivo.

Sin embargo, la implementación de una política de red compleja y reactiva con herramientas estáticas como las reglas de un firewall requiere que los administradores de red configuren de forma independiente múltiples componentes diferentes, incluidos los

swtiches, los servidores de administración y numerosos scripts ad hoc aumentando el riesgo a estar propenso a errores y dificultando la gestión de la red.

- Simulación de la red de arquitectura SDN

El despliegue de nuevas características en la red como habilitación de protocolo BGP, despliegue de una nueva VLAN o bloqueo de puertos específicos; se realizaron desde el controlador SDN instalando nuevas tablas de flujos en los switches de forma intuitiva con la interfaz AP OFM. El tiempo aproximado fue de 50 mseg por swtich SDN.

SDN simplifica significativamente el despliegue de estos tipos de políticas. SDN permitió implementar este tipo de políticas de red reactiva sin configuraciones complejas de red de bajo nivel que las redes tradicionales requerirían.

ARQUITECTURA DE RED	TIEMPO DE IMPLEMENTACIÓN POR DISPOSITIVO
TRADICIONAL	5 minutos aprox.
SDN	50 mseg aprox.

Tabla 5.9: Tiempo de despliegue de configuración.
Fuente: Elaboración propia.

Según la tabla 5.9, SDN facilito la creación y administración de tareas de automatización y despliegue de nuevas configuraciones, añadiendo que uno de los beneficios de SDN la visibilidad de la red desde una ubicación centralizada

CAPITULO VI

PRESUPUESTO BÁSICO ESTIMADO PARA IMPLANTACIÓN DE UNA RED SDN

6.1. DISPOSITIVOS DE COMUNICACIÓN COMPATIBLES CON OPENFLOW

Un switch OpenFlow puede ser software o dispositivo de hardware que reenvía paquetes en un entorno de redes definidas por software (SDN). Los switches OpenFlow están basados en el protocolo OpenFlow o son compatibles con OpenFlow.

No se requieren switch SDN especiales para implementar redes definidas por software, se recomienda usar switches híbridos. Esto significa que la nueva red debe diseñarse para admitir protocolos SDN en entornos de swithces físicos y virtuales, y ser compatible con las redes tradicionales.

Una red híbrida ofrece la ventaja de una migración gradual desde su arquitectura de red actual hacia una red defina por software pura. La red hibrida ofrece la capacidad de aprovechar la red telemática existente de Layer 2 y Layer 3 sin tener que reemplazar la red. La red hibrida busca obtener los beneficios de SDN, que incluyen mejor aprovisionamiento, flexibilidad de red, calidad de servicio (QoS) y programabilidad durante la migración de SDN.

La mayoría de los proveedores de SDN tienen arquitecturas y productos que admiten una migración gradual de las redes tradicionales a SDN, que incluyen Cisco, HP, IBM, Juniper, Brocade, Avaya, VMware, Big Switch, ADARA Networks, Embrane, Pertino, Pluribus Networks, Vello Systems, entre otros.

Según la **ONF (2018)**, los dispositivos aprobados para operar en una red SDN son los siguientes:

Fecha de aprobación	Nombre del Fabricante	Nombre del Producto	Versión del firmware
2018-05	Allied Telesis, Inc.	AT-XS900MX 10 Gigabit Layer 3 stackable switches	AlliedWare Plus v5
2018-04	DASAN Network Solutions Inc.	V6848XG 48 ports 10GL3 Switch	2.14 0050
2018-04	KTNF Inc.	K2632SI System with 36 ports 10GbE Switch and Intel Xeon Server	n2os-0.03.02-onie-installer-fm10k-20180315

2018-03	Allied Telesis, Inc.	AT-x310 Layer 3 stackable Access Switches with 1G Uplinks	AlliedWare Plus v5
2018-03	Allied Telesis, Inc.	AT-x230 Layer 3 Gigabit Edge switches with 1G SFP uplinks	AlliedWare Plus v5
2018-03	Allied Telesis, Inc.	AT-SBx908 GEN2 High Capacity Stackable Layer3+ Modular Switch with 10G/40G/100G XEMs	AlliedWare Plus v5
2018-02	Allied Telesis, Inc.	AT-x550 10 Gigabit Layer 3 Stackable Switches with 40G Uplinks	AlliedWare Plus v5
2018-01	Allied Telesis, Inc.	AT-x510 Gigabit Layer 3 Stackable Switches with 1/10G Uplinks	AlliedWare Plus v5
2018-01	Netvision Telecom Inc.	NetVOF-48X OpenFlow Switch of Netvision Telecom	(cl.ver1)2.5.0-199c587-201501081931-build
2017-12	Allied Telesis, Inc.	AT-x930 Advanced Gigabit Layer 3 Stackable Switches with 10G and 40G Uplinks	AlliedWare Plus v5
2017-12	Ruijie Networks Co., Ltd	RG-N18012 Ruijie Newton 18000 Switch Series leads the industry in supporting cloud data center with a broad spectrum of specialized campus network features.	RGOS 11.0
2017-12	Ruijie Networks Co., Ltd	RG-S8612E Ruijie RG-S8600E Switch Series is industry leads the industry in supporting cloud data center with a broad spectrum of specialized campus network features.	RGOS 11.0

2017-10	Maipu Communication Technology Co., Ltd.	SDN Router NSR self-controllable router uses the domestic multi-core processor and supports IPv4, IPv6, MPLS and other network protocol.	H011
2017-10	Maipu Communication Technology Co., Ltd.	SDN Switch MyPower S12800 series switch	20
2017-10	Maipu Communication Technology Co., Ltd.	SDN Switch MyPower S5820 series new-generation data center-class access switch	10
2017-10	Maipu Communication Technology Co., Ltd.	SDN Router MP2900X supports the 4G access of the 4-way GSM fully	H011
2017-09	Nanjing Balance Network Technology Co., Ltd	BLC-A40-S600 BLC-A40-S600 SDN Switch	P050R006
2017-09	DASAN Network Solutions Inc.	V6748XG OpenFlow enabled 48 ports 1G/10G L3 Switch	2.14 0050
	Technologies Co., Ltd.	DPX19000-A6 Core Switch	
2016-12	HUAWEI Technologies Co., Ltd.	S5720-12TP-LI-AC L2 Ethernet Switch with 8 Ethernet 10/100/1000 Base - T,2 GE SFP and 2 dual-purpose 10/100/1000Base-T or SFP	V2R10C00
2016-12	HUAWEI Technologies Co., Ltd.	S628X-E L2 Ethernet Switch with 24 Ethernet 10/100/1000 Base-T, 4 10GE SFP+	V2R10C00
2016-12	Hangzhou H3C Technologies Co., Ltd.	H3C S6820-4C L3 Ethernet Switch	142
2016-12	Hangzhou H3C	H3C S5560X-54C-EI	ESS 1100

	Technologies Co., Ltd.	L3 Ethernet Switch with 48 10/100/1000BASE-T, 4 SFP Plus Ports and 1 sub-slot	
2016-12	Hangzhou H3C Technologies Co., Ltd.	S9810 DataCenter Cloud High Density Switch	CMW710
2016-12	Hangzhou H3C Technologies Co., Ltd.	S12508X-AF DataCenter Cloud High Density Switch	CMW710
2016-12	Hangzhou H3C Technologies Co., Ltd.	SR8808-X H3C SR8808-X Core Router	CMW710
2016-12	Ruijie Networks Co., Ltd.	RG-S2910-24GT4SFP-UP-H, RG-S2910-24GT4XS-E, RG-S2910-48GT4XS-E, RG-S2910C-24GT2XS-HP-E, RG-S2910C-48GT2XS-HP-E Ruijie RG-S2910 Switch Series is a collection of next-gen Gigabit switches architected for superior security, high performance and outstanding energy efficiency.	RGOS 11.4
2016-12	Ruijie Networks Co., Ltd.	RG-S5750C-28GT4XS-H, RG-S5750C-28SFP4XS-H, RG-S5750C-48GT4XS-H Ruijie RG-S5750-H Switch Series is a collection of next-gen multiservice switches, offering remarkable performance and enhanced security.	RGOS 11.4
2016-06	Hangzhou H3C Technologies Co., Ltd.	S10508 H3C S10508 Core Switch	CMW710
2016-06	Hangzhou H3C Technologies Co., Ltd.	S10508 H3C S7560E Core Switch	CMW710

2016-05	Ruijie Networks Co., Ltd.	RG-S6220-48XS6QXS-H The RG-S6220 series is a collection of 10GE data center switches, offering non-blocking, unified and virtualized switch performance	RGOS 11.0
2016-05	Hangzhou H3C Technologies Co., Ltd.	H3C MSR 56-60 Open Multi-Series Router with dual boards and 6 HMIM Slots	7.1
2016-02	Ruijie Networks Co., Ltd.	RG-N18010 Ruijie Newton RG-N18010 Switch is industry leading in supporting cloud data center with a broad spectrum of specialized campus network features.	RGOS 11.3
2016-02	Hangzhou H3C Technologies Co., Ltd.	H3C S6800-2C L3 Ethernet Switch with 2 QSFP Plus Ports and 2 Slots	142
2016-02	Hangzhou H3C Technologies Co., Ltd.	H3C S6800-32Q L3 Ethernet Switch with 32 QSFP Plus Ports	142
2016-02	Hangzhou H3C Technologies Co., Ltd.	H3C S6800-4C L3 Ethernet Switch with 4 Slots	142
2016-02	Hangzhou H3C Technologies Co., Ltd.	H3C S6800-54QT L3 Ethernet Switch with 48 10GBASE-T Ports and 6 QSFP Plus Ports	142
2015-12	Huawei Technologies Co., Ltd.	S5720-52X-SI-AC L3 Ethernet Core Switch with 48 10/100/1000BASE-T, 4 10GE SFP+	V2R8C06
2015-12	Huawei Technologies Co., Ltd.	S7706 L3 Ethernet Switch with 48	V2R8C06

		10/100/1000BASE-T card	
2015-12	Huawei Technologies Co., Ltd.	S9306 L3 Ethernet Switch with 48 10/100/1000BASE-T card	V2R8C06
2015-11	Huawei Technologies Co., Ltd.	S6720-54C-EI-48S L3 Ethernet Switch with 48 x GE SFP/10 GE SFP+2 QSFP+ 4 x40GE QSFP+	V2R8C06
2015-11	Huawei Technologies Co., Ltd.	S6720-54C-EI-48S L3 Ethernet Switch with 28 10/100/1000BASE-T, 4 Combo SFP Ports and 4 10GE SFP+	V2R8C06
2015-11	Huawei Technologies Co., Ltd.	S12708 L3 Ethernet Core Switch with 48 10/100/1000BASE-T card	V2R8C06
2015-09	Digital China Networks, Ltd.	DCRS-7604 OpenFlow/L3 Ethernet Switch	7.4.3.0(R0001.0081)
2015-09	Hangzhou H3C Technologies Co., Ltd.	S5130-54QF-HI L3 Ethernet Switch with 48 10/100/1000Base-T,4 SFP Plus Ports and 1 sub-slot	ESS 1100
2015-09	Hangzhou H3C Technologies Co., Ltd.	H3C S6800-54QF L3 Ethernet Switch with 48 SFP Plus Ports and 6 QSFP Plus Ports	142
2015-09	Huawei Technologies Co., Ltd.	CE6851-48S6Q-HI Highly versatile, SDN-ready Ethernet switches	V100R006
2015-09	ZTE Corporation	ZXR10 M6000-S SDN Switch	CTN9000-E V3.00.10(2.20.1)

Tabla 6.1: Lista de dispositivos certificados por la ONF.
Fuente: ONF (2018)

6.2.PRESUPUESTO ESTIMADO DE DISPOSITIVOS SDN CON EL PROVEEDOR HP

Para el presente presupuesto se utilizó como base los equipos del proveedor HP por sus características técnicas y brindar la posibilidad de utilizar aplicaciones de tercer. Según **HP COMPANY (2018)**, la tienda de aplicaciones HP SDN ofrece a los proveedores de software independientes una manera fácil de llevar soluciones creativas al mercado, permitiendo a los administradores de TI resolver sus desafíos de red únicos a través de estas aplicaciones.

El estándar OpenFlow v1.3 incluye especificaciones para las interacciones híbridas entre el tráfico OpenFlow y el tráfico que no es de OpenFlow, para permitir una migración temprana de SDN. OpenFlow v.1.3 especifica dos tipos de switches compatibles con OpenFlow: OpenFlow puro e híbrido. Los switches híbridos OpenFlow admiten OpenFlow y la tecnología de conmutación de una red tradicional. El proveedor HP utiliza hardware de red que soporta la posibilidad de ejecutarlos en modo híbrido.

6.2.1. CONTROLADOR

El HP Open SDN Ecosystem incluye un HP SDN Developer Kit y un HP SDN App Store con HP y aplicaciones y servicios SDN desarrollados por terceros para proporcionar soluciones SDN listas para la empresa para el centro de datos, el campus y la sucursal.

Según **HP COMPANY (2018)**, Virtual Application Networks (VAN) SDN Controller de HP proporciona un punto de control unificado en una red compatible con SDN, lo que simplifica la administración, el aprovisionamiento y la orquestación. Esto permite la entrega de una nueva generación de servicios de red basados en aplicaciones y proporciona interfaces de programación de aplicaciones abiertas (API) que permiten a los desarrolladores crear soluciones innovadoras para vincular dinámicamente los requisitos del negocio con la infraestructura de red a través de programas Java personalizados o interfaces de control RESTful de propósito general.

El controlador HP VAN SDN está diseñado para funcionar en entornos de campus, centro de datos o proveedor de servicios y brinda las siguientes características principales:

- Procesamiento de flujo proactivo
- Procesamiento de flujo reactivo
- Interfaz gráfica de usuario (GUI)
- API hacia el norte
- API nativas
- Arquitectura escalable
- Alta disponibilidad
- Seguridad del controlador
- Módulo de servicio de enlace
- Módulo de servicio de topología
- Módulo de servicio del administrador de nodo
- Interfaz de control de OpenFlow

- Procesamiento flexible de paquetes

6.2.2. SWITCH HP 2920 24G

Según **HP COMPANY (2018)**, la Serie 2920 ofrece seguridad, escalabilidad y facilidad de uso para redes empresariales de borde, SMB y sucursales. El switch HP 2920 admite apilamiento modular, 10 GbE, RIP y acceso al enrutamiento OSPF, nodo tunelizado, PoE +, ACL, sFlow, QoS robusto e IPv6. Está listo para la red definida por software con API REST y soporte OpenFlow.

Principales características:

- Serie de conmutadores Aruba Layer 3 con apilamiento, enrutamiento estático y RIP, IPv6, ACL y sFlow para una mejor experiencia de campus móvil primero
- Enlaces ascendentes modulares de 10 GbE y fuentes de alimentación actualizables para hasta 1440 W PoE +
- Soporte OpenFlow

6.2.3. SWITCH HP 3800-24G-2XG

Según **HP COMPANY (2018)**, la serie de conmutadores HPE 3800 es una familia de nueve conmutadores Gigabit Ethernet totalmente gestionados disponibles en modelos de 24 puertos y 48 puertos, con o sin PoE + y con enlaces ascendentes SFP + o 10GBASE-T.

Principales características:

- Swtiches apilables layer 3 completamente gestionados
- Arquitectura de baja latencia altamente resistente
- SFP +, 10GBASE-T, PoE + y apilamiento modular
- Tecnología de apilamiento mallado altamente resistente
- Soporte OpenFlow

6.2.4. SWITCH HP SWITCH 5400ZL SERIES

Según **HP COMPANY (2018)**, la serie HP Switch 5400zl consta de los conmutadores inteligentes más avanzados de la línea de productos HP ProCurve. La serie 5400zl incluye chasis de 6 ranuras y 12 ranuras y módulos zl y paquetes asociados. La base de todos estos conmutadores es ProVision ASIC programable y especialmente diseñado que permite implementar las funciones de red más exigentes, como la calidad de servicio (QoS) y la seguridad, de una manera escalable y granular. Con interfaces 10/100, Gigabit y 10-Gigabit, PoE + integrado en puertos 10/100 y 10/100 / 1000Base-T, y una selección de factores de forma, los switches 5400zl ofrecen una excelente protección de la inversión, flexibilidad y escalabilidad, así como también como facilidad de implementación, operación y mantenimiento.

Principales características:

- Core, distribución y capa de acceso avanzado
- Capa 2 a 4 y conjunto de características de borde inteligente
- Rendimiento y seguridad de clase empresarial
- Soporte OpenFlow
- Conectividad escalable de 10/100/1000 y 10 GbE

6.2.5. EQUIPOS TERMINALES

Los equipos terminales tales como los computadores, impresoras, laptops, teléfonos móviles, entre otros; no necesitan características adicionales para integrarse a la red SDN.

6.2.6. PRESUPUESTO ESTIMADO

La cantidad de equipos se calcularon de acuerdo con la topología planteada para la simulación.

Las tablas de precio siguientes se estimaron a base de los precios de **ITPRICE (2018):**

DISPOSITIVO	FABRICANTE	MODELO	Precio por Unidad (USD)	Cantidad	Total (USD)
ACCESO	Hewlett-Packard (HP)	HP 2920-24G	$2,719.00	40	$108,760.00
DISTRIBUCIÓN	Hewlett-Packard (HP)	HP 3800-24G-2XG	$13,026.00	10	$130,260.00
CORE	Hewlett-Packard (HP)	HP 5406 zl switch	$25,335.00	2	$50,670.00
CONTROLADOR	HP VAN Open SDN	Aruba VAN SDN Ctrl HA E-LTU	$9,990.00	2	$19,980.00
TOTAL					$309,670.00

Tabla 6.2: Presupuesto estimado para la red telemática.
Fuente: ITPRICE (2018)

DISPOSITIVO	FABRICANTE	MODELO	Precio por Unidad (USD)	Cantidad	Total (USD)
ACCESO	Hewlett-Packard (HP)	HP 2920-24G	$2,719.00	4	$10,876.00
DISTRIBUCIÓN	Hewlett-Packard (HP)	HP 3800-24G-2XG	$13,026.00	1	$13,026.00
CONTROLADOR	Servidor HP con Opendaylight	HP BL660c Gen9 Intel Xeon E5-4610v3	$2,500.00	1	$2,500.00
TOTAL					$25,402.00

Tabla 6.3: Presupuesto estimado para un ambiente de laboratorio SDN.
Fuente: ITPRICE (2018)

6.3.BENEFICIO ECONÓMICO

La centralización de la inteligencia y la lógica y eliminarla de los dispositivos de red permite la reducción del costo de los switches y, por lo tanto, el costo total del equipo de

red. La reducción de costos ha sido el mejor impulsor detrás del movimiento hacia la tecnología SDN.

6.3.1. CAPEX

Según **MURAT, ARJAN (2017)**, algunos de los factores clave que afectan el potencial de reducciones de CAPEX incluyen:

- Dispositivos de red más simples: en una red SDN. Los dispositivos serán cajas blancas más simples y más baratas. Esto debería reducir el costo de cada dispositivo para los operadores de red.
- Componentes adicionales: en un escenario SDN, la red tendrá elementos adicionales que una red tradicional no tiene. Estos elementos incluyen el hardware del controlador y las licencias de software del controlador (si no se utiliza un software de código abierto). Estos elementos, sin embargo, contribuirán al CAPEX de la red.
- Dimensionamiento de red: este factor se refiere a la capacidad de la red a través del número de dispositivos de red necesarios para manejar la carga de la red. Como los controladores de red pueden tener una vista de red global en caso de SDN, esto permite una mejor utilización de los recursos de la red mediante algunos métodos, como el equilibrio de carga. Por lo tanto, es posible que no sea necesario aprovisionar en exceso la red, lo que puede reducir los gastos de capital

6.3.2. OPEX

Según **MURAT, ARJAN (2017)**, SDN promete reducir algunos de los componentes principales de OPEX por medio de sus diversas características:

- Costos relacionados con la energía: En una red SDN, los switches no tendrán un plano de control incorporado, que consume la mayor parte de la energía total que necesita un switch.

  Además, dado que SDN permite una optimización del tráfico más eficiente en los dispositivos de red, esto reduce la cantidad total de dispositivos necesarios. Esto conducirá a un menor costo de energía. Sin embargo, un controlador independiente contribuirá al consumo de energía. A pesar de esta contribución del (de los) controlador (es), se espera que el consumo total de energía sea menor en el escenario SDN.

- Costos de mantenimiento: SDN crea un entorno de red homogéneo con respecto al hardware y el software. No hay ningún caso en el que los diferentes dispositivos dependientes del proveedor deban gestionarse y mantenerse de manera independiente.

  La administración de software se vuelve más fácil en comparación con el caso de red tradicional, ya que no se usarán varias versiones de software. Es probable que también se vean efectos similares para la administración de seguridad.

- Costos de reparación: SDN proporciona mejores oportunidades de prueba, identifica errores, y así sucesivamente antes de llegar al tráfico de producción real. Esta característica debería reducir el costo de las reparaciones en una red SDN. Sin embargo, el problema del punto único de falla para el (los) controlador (es) es un problema en SDN.

  En caso de tales problemas, la reparación de la red también sería demasiado costosa. Este problema puede superarse mediante arquitecturas de red distribuidas. Además, las fallas en los dispositivos de red pueden desestabilizar las operaciones de la red. Los problemas relacionados con el software se pueden solucionar de forma remota sin tocar los dispositivos de red, ya que estos dispositivos son simples y SDN está centrado en el software.

- Costos de provisión de servicios: Se espera que el costo de provisión de servicios en el escenario SDN sea más bajo debido a la configuración automatizada de la red. En una red tradicional, proporcionar un servicio a un cliente puede requerir ciertas tareas como la configuración, el cambio, etc. Estas tareas pueden ocasionar más errores debido a las configuraciones manuales por parte del personal, lo que a su vez puede causar un mayor tiempo de inactividad de la red. Además, el personal para manejar estas tareas puede ser difícil de encontrar, costoso y difícil de retener. Todos estos factores resultan en un menor costo de provisión del servicio, por lo tanto, menos OPEX, en el caso de SDN.

6.4. CASO DE ESTUDIO

Según **SDXCENTRAL (2015**), el Sistema de Escuelas del Condado de South Washington, con un solo profesional de TI para administrar y configurar sus redes, necesitaba una forma más fácil de mantener una seguridad estricta y administrar redes cableadas e inalámbricas en 31 sitios. Después de revisar las propuestas para la costosa seguridad basada en hardware, el distrito de Minnesota eligió una solución HP SDN y ahorró al distrito cientos de miles de dólares en costos de instalación y mantenimiento. Los costos iniciales totalizaron menos de $ 200,000, en comparación con aproximadamente $ 2 millones para la seguridad basada en hardware. El distrito pudo implementar la solución en cerca de 400 conmutadores HP habilitados para SDN en menos de una hora y a una fracción del costo de los dispositivos de seguridad de red basados en hardware.

Las amenazas de seguridad ahora se detectan fácilmente puerto por puerto, en lugar de depender de los firewalls en el perímetro de la red. Según **SDXCENTRAL (2015**), el distrito atrapa más de 100.000 solicitudes DNS maliciosas de un total de 22 millones de solicitudes de DNS cada día escol

CAPITULO VII

LÍNEAS DE INVESTIGACIÓN EN SDN

7.1.FUTUROS DESAFÍOS EN LAS REDES SDN

La arquitectura de red SDN permite a los investigadores desarrollar diversas aplicaciones tales como balanceo de carga, virtualización de red, control de acceso dinámico en redes corporativas, movilidad de máquinas virtuales, etc. Según **SHAILENDRA, MOHAMMED (2017)**, para resolver los problemas de redes como, costos, recursos técnicos, plano de control separado para configuraciones, visibilidad descentralizada de dispositivos de red, VLAN separadas para cada red, ingeniería de tráfico compleja, ancho de banda por cada servicio, es necesario probar y desarrollar diversas aplicaciones sobre SDN.

SDN implica que los sistemas están controlados por aplicaciones de programación. La arquitectura de red SDN es un diseño que indica que es cambiante, inteligente y versátil, tratando de responder a la alta transmisión de datos, que es la naturaleza dinámica de las aplicaciones de hoy en día. La convención OpenFlow se puede usar como parte de los avances de SDN. El conocimiento de la red se reúne en controladores de red definidos por software de programación, lo que mantiene una perspectiva global de la red. Se organiza automáticamente, es decir, los activos del sistema rápidamente a través de programas SDN dinámicos y mecanizados. Esto impulsa a destacar la interoperabilidad, el desarrollo y arreglos más adaptables y con mayor conocimiento de la aplicación.

Según **SHAILENDRA, MOHAMMED (2017),** SDN ha logrado allanar el camino hacia una red de próxima generación, generando un entorno de investigación y desarrollo innovador, promoviendo avances en varias áreas: diseño de plataforma de conmutación y controlador, evolución de escalabilidad y rendimiento de dispositivos y arquitecturas, promoción de seguridad y confiabilidad.

7.2.LÍNEAS DE INVESTIGACIÓN A DESARROLLAR

Según **SHAILENDRA, MOHAMMED (2017)**, La arquitectura de red SDN presenta los siguientes retos:

1. ¿Cómo responde el controlador SDN ante las decisiones subjetivas y contradictorias?
2. Identificar las propiedades de los controladores SDN y su impacto en la técnica de toma de decisiones complejas.
3. Identificación del mejor controlador SDN sea de código abierto o propietario.
4. El controlador SDN debe actuar como activo-en espera y, en el caso de que uno de los controladores esté inactivo, todas las tablas de flujo deben ir al controlador de respaldo, para que las operaciones de tráfico no se alteren.

5. Equilibrio de carga multitrayecto dinámico, en caso de carga de congestión puede ser compartida.
6. Manejar una gran cantidad de datos en la red requiere una CPU y memoria altas en el lado del controlador para que todas las solicitudes se manejen sin demoras, por lo cual se necesita buscar una forma de minimizar los recursos de hardware.
7. Dependencia de cómo y como afecta la cantidad de flujos que el controlador SDN puede manejar cada vez que configura un flujo cuando se inicia un nuevo switch.
8. Los switches de hardware de Openflow envían flujo para la toma de decisiones al controlador SDN y de esta forma pueden llegar miles de solicitudes de flujo, lo que puede generar latencia
9. La ubicación del controlador es muy importante, ya que puede causar retrasos si no se coloca correctamente.
10. El problema de la interoperabilidad en SDN es uno de los principales problemas. Es necesario tener una interfaz para la comunicación entre SDN y el plano de control no SDN como MPLS, esto aumentará la escalabilidad de la red que se está ejecutando.
11. Desde el punto de vista de la seguridad, debe haber alguna lista de control de acceso ACL.

7.2.1. RETOS Y OPORTUNIDADES PARA LA SEGURIDAD EN SDN

Según **AKRAM ed al. (2014)**, la seguridad en las redes SDN plantea importantes desafíos porque su aspecto programable presenta un conjunto complejo de problemas para hacer frente. Se espera que el creciente número de ataques de DDoS y malware, spam y actividades de phishing cambien la dinámica en torno a la protección de las infraestructuras SDN. En este contexto, las redes inalámbricas móviles son más vulnerables que las redes cableadas fijas ya que los canales inalámbricos de difusión permiten fácilmente interceptar e interceptar mensajes (es decir, vulnerabilidad de canales). Además, las redes móviles ad-hoc incurren en desafíos de seguridad más complejos debido a su falta de infraestructura (por ejemplo, servidores de seguridad), lo que hace que las soluciones de seguridad clásicas no sean viables. Los enfoques de seguridad tradicionales requieren un tiempo de inactividad para organizar los cambios de topología mientras se reconfigura la red, se inserta una nueva configuración de seguridad y se activan y depuran varios servicios de seguridad, por tal razón se están buscando soluciones nuevas bajo la arquitectura de redes SDN.

7.2.2. SDN PARA REDES BASADAS EN LA NUBE

La introducción y el despliegue de servicios basados en la nube se han convertido en una solución importante que ofrece a las empresas un modelo de negocio rentable. Según **AKRAM ed al. (2014)**, muchas funciones de red en la nube tienen características y requisitos de rendimiento extremos, lo que ha creado nuevos desafíos tales como servidores y virtualización de red, nubes móviles y seguridad. Estos deben abordarse con virtualización de red inteligente, procesamiento de paquetes de alta velocidad y equilibrio de carga. SDN se puede ver como una tecnología nueva y complementaria a la

virtualización, que está lista para enfrentar los desafíos de las implementaciones en la nube y en la red a través de la red.

7.2.3. VISIBILIDAD DE RED Y DESAFÍOS DE GESTIÓN CON SDN

SDN presenta nuevas capacidades de administración y monitoreo que pueden mejorar el rendimiento y reducir los cuellos de botella en la red. Según **AKRAM ed al. (2014)**, aunque las herramientas de supervisión de SDN pueden ser poderosas en topologías de red pequeñas y medianas, la depuración, la resolución de problemas, el monitoreo y el cumplimiento de la seguridad son tareas muy difíciles en SDN distribuidas.

Las herramientas de supervisión de SDN deberían proporcionar una capacidad proactiva que aproveche la flexibilidad y la programabilidad de SDN para crear una supervisión elástica de la red. Deben proporcionar lenguajes de gestión declarativos de alto nivel para garantizar la coherencia de los estados de la red y detectar fallas en tiempo real. Estos lenguajes deberían implementar una amplia gama de políticas de red para diferentes tareas de administración (es decir, QoS, VLAN, aislamiento de red, segmentación, etc.) para evitar errores a medida que surgen, independientemente de la implementación de cualquier controlador específico. Por lo tanto, las herramientas existentes deben ampliarse para admitir la conciencia del servicio para mapear interactivamente los flujos en los servicios, identificar los flujos seleccionados de todas las listas de flujos y supervisar el estado de los flujos seleccionados.

7.2.4. CONVERGENCIA DE RED Y QOS

Según **AKRAM ed al. (2014),** con la necesidad de flexibilidad y la coexistencia eficiente de múltiples servicios (tales como, VoIP, video y datos) dentro de un único switch, las empresas deben hacer frente a una gran demanda de Calidad de Servicio (QoS), Calidad de Experiencia (QoE), robustez, facilidad de modificación y/o actualización, seguridad y privacidad. Los proveedores deben ser capaces de crear divisiones de red virtuales en el mismo equipo de red al mismo tiempo que aseguran el rendimiento y el aislamiento requeridos en los diferentes servicios para habilitar la QoS basada en la aplicación.

7.3.LÍNEAS DE INVESTIGACIÓN A DESARROLLAR

Los principales proyectos de investigación se centran en la arquitectura SDN para desarrollar y presentar un mapa lógicamente centralizado de la red. Se espera que la próxima generación de redes SDN se beneficie no solo de la simplicidad de la implementación, sino también de mantener y actualizar las aplicaciones de forma rápida y sencilla.

Según **AKRAM ed al. (2014)**, aunque OpenFlow promete un mecanismo de transmisión de flujo flexible, abierto y dinámico, también plantea una serie de desafíos en términos de virtualización de red, gestión de movilidad y operación, lo que requerirá la atención coordinada de la comunidad investigadora para su éxito y amplia aceptación.

CAPITULO VIII

CONCLUSIONES

- El simulador de red Mininet permitió diseñar de una red LAN bajo la arquitectura SDN para la Red Telemática de la UNMSM en un entorno de simulación usando el controlador Opendaylight.

- Se logró observar el potencial de las redes SDN incluso en un entorno de red virtualizado. Mininet permitió ejecutarlas aplicaciones, módulos y comandos del sistema Linux desde los hosts virtuales sin inconvenientes.

- Se pudo poner en evidencia que el controlador Opendaylight es capaz de administrar todos los dispositivos de red que tengan habilitado Openflow, además el controlador tiene la total visibilidad de la red de manera centralizada permitiendo una gestión unificada.

- Las redes definidas por software permitirían a la Red Telemática habilitar la programación de la red mediante distintas APIs de programación. En el caso particular del controlador Opendaylight, se utilizó el lenguaje de programación XML a través de RESTCONF. De esta manera, no fue necesario la configuración a nivel local del dispositivo de la red a través de línea de comandos permitiendo la posibilidad de cambiar los dispositivos de red actuales (con un comportamiento predefinido por el fabricante) por dispositivos que permiten ser programados según las necesidades del administrador.

- Los resultados obtenidos en las pruebas conectividad nos permiten concluir que el controlador Opendaylight puede manejar tráfico TCP y UDP. Una vez establecida la conexión y la instalación de las tablas de flujo en los switches, los tiempos de respuesta son menores con respecto a los primeros paquetes enviados.

- SDN brinda la oportunidad a los investigadores de desarrollar las líneas de investigación en el campo de la automatización de la red, seguridad de las redes de forma proactiva, convergencia en la red y desarrollo de aplicaciones proactivas de QoS.

- La configuración de los dispositivos de red por CLI no es escalable y presentan inconvenientes de integración debido a la diversidad de sintaxis exclusiva de cada fabricante. Por otra parte, SDN proporciona APIs de programación dinámicos y fluidos entre el software y la infraestructura de red. SDN brindaría a la red telemática la posibilidad de mantener la integración a medida que la infraestructura de red evoluciona, reduciendo la complejidad operativa y costes mediante la reducción de la complejidad al usuario, y haciendo un uso más eficiente de los recursos de red existentes.

- La automatización de la gestión y provisión de la red, a través de SDN, es la siguiente fase en la virtualización de la infraestructura de tecnologías de la información y las telecomunicaciones ya que permite crea una red inteligente mucho más abierta, flexible, escalable y reprogramable.

CAPITULO IX

OBSERVACIONES

- El módulo L2-Switch del controlador Opendaylight permitió la instalación de tablas de flujos en los switch OVS de manera automatizada y centralizada, soportando el tráfico unicast y broadcast de la red simulada. El controlador instaló cada flujo de paquetes de origen a destino en cada switch Openflow. Los protocolos talos como Spanning Tree (STP) u OSFP (protocolo de enrutamiento) no determinaron las rutas que los paquetes siguieron a través de la red.

- El controlador Opendaylight permitió ejecutar el protocolo de enrutamiento BGP para la comunicación con redes tradicionales. La interoperabilidad del protocolo BGP entre el controlador y el Router tradicional fue transparente, lo cual permitiría la integración de la Red Telemática SDN con redes que aún no adopten la arquitectura SDN, como Internet.

- El controlador Opendaylight aún no tiene un API Northbound estandarizado, lo cual dificulta la interoperabilidad de las aplicaciones SDN en otros controladores SDN. En este caso particular, se utilizó RESTCONF para la comunicación con el controlador, sin embargo, se pueden usar otros protocolos.

- La introducción de la tecnología SDN a la Red Telemática de la UNMSM requerirá que los administradores de red tengan conocimiento en programación, automatización y conocimiento en distintos lenguajes de programación para utilizar los servicios de gestión basados en software.

- El despliegue de redes SDN en ambientes de producción aún se encuentra en pleno desarrollo de la tecnología y de estandarización para garantizar su calidad e interoperabilidad. El controlador Opendaylight aún se encuentra en constante cambio, su mayor uso es en ambientes de investigación o pruebas.

- El futuros estudios e investigaciones sobre las aplicaciones de una red SDN permitirán la creación de aplicaciones que soliciten y modifiquen dinámicamente los servicios proporcionados por la infraestructura de red, sin necesidad de intervención humana, pudiendo la red informar de su nuevo estado a las aplicaciones.

CAPITULO X

RECOMENDACIONES

- Se recomienda una PC de hardware robusto para simular un entorno SDN debido a que el controlador y Mininet consumen elevados recursos de hardware.

- La presente tesis fue descrita sobre una red virtualizada, por cual se recomienda para futuros estudios disponer de switches físicos con soporte de Openflow para implementar entornos de pruebas y de virtualización que permiten comprobar el rendimiento del controlador SDN Opendaylight en un entorno real. Así como realizar la evaluación usando diferentes tipos de trafico en la red.

- Se recomienda realizar pruebas con otros controladores SDN para comparar los diferentes resultados y determinar el controlador óptimo de acuerdo con las necesidades de la Red Telemática.

- Para la transición entre las redes SDN y redes tradicionales, se recomienda el despliegue de switches híbridos que operen en ambas arquitecturas de red para una mejor transición entre ambas arquitecturas.

REFERENCIAS BIBLIOGRÁFICAS

[1]. Akram, H. Aniruddha, G. Pascal, B. Douglas, C. Gayraud, T. (2014). *Software-defined Networking: Challenges and Research Opportunities for Future Internet*. Diciembre 2014, de Vanderbilt University.

[2]. Big Switch Inc. (2011). *Can SDN be based on CLI?*. 2011, de Big Switch. Recuperado de: https://www.bigswitch.com/blog/2011/11/21/can-sdn-be-based-on-cli

[3]. Big Switch Inc. (2012*). Openflow MythBusting by Google*. 2012, de Big Switch Inc. Recuperado de: https://www.bigswitch.com/blog/2012/04/30/openflow-mythbusting-by-google

[4]. Bruno, A. y Jordan, S. (2017). *CCDA 200-310 Official Cert Guide*. Estados Unidos: Cisco Press.

[5]. Bhambri, A. (2011). *Looking for Data Scientists from Within – Start with Marketing*. Julio 2011, de Dataversity. Recuperado de: http://www.dataversity.net/looking-for-data-scientists-from-within-start-with-marketing/|

[6]. Case, J., Mundy, R., Partain, D. y Stewart, B. (2012). *Introduction and Applicability Statements for Internet Standard Management Framework.* diciembre, 2012, de IETF RFC3410. Recuperado de: https://tools.ietf.org/html/rfc3410

[7]. Cisco System. (2017). *The Zettabyte Era: Trends and Analysis*. 2017, de Cisco System. Recuperado de: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/vni-hyperconnectivity-wp.pdf

[8]. Cuba, G. y Becerra M. (2015). *Diseño e implementación de un controlador sdn/openflow para una red de campus académica* (Tesis de pregrado) Pontifica Universidad Catolica del Peru, Lima, Perú.

[9]. Dandekar, A. y Kharade, R. (2015). *Implementing a low cost SDN ecosystem in LAN*. India Conference (INDICON). De IEEE Xplore Base de datos.

[10]. Enns, R., Bjorklund, M., Schoenwaelder, J. y Bierman, A. (2011). *Network Configuration Protocol (NETCONF)*. junio, 2011, de IETF RFC6241. Recuperado de: https://tools.ietf.org/html/rfc6241

[11]. Errnado, I. (2013). L*a solución que necesita la red se llama Software Defined Netowk*. Data Center Dinamics - Focus, 12, pp.26-27. Recuperado de: http://content.yudu.com/Library/A20af7/DataCentreDynamicsIs/resources/27.htm

[12]. España, N. (2016). *Diseño y simulación de una red definida por software (sdn)* (Tesis de pregrado). Universidad Central del Ecuador, Quito, Ecuador.

[13]. GARTNER. (2016). *Predicts 2017: Enterprise Networks and Network Services*. Noviembre, 2016, de GARTNER. Recuperado de: https://www.gartner.com/doc/3518017/predicts--enterprise-networks-network

[14]. GNS3 (2017). Recuperado de: https://www.gns3.com/

[15]. Goransson, P., Black, P. y Culver, T. (2016). *Software Defined Networks, Second Edition: A Comprehensive Approach*. Estados Unidos: Morgan Kaufmann.

[16]. Haleplidis, E., Pentikousis, K., Denazis, S., Hadi, J., Meyer, D. y Koufopavlou, O. (2015). *Software-Defined Networking (SDN): Layers and Architecture Terminology*. Enero, 2015, de IETF RFC7426. Recuperado de: https://tools.ietf.org/html/rfc7426

[17]. Hewlett-Packard Development Company (HP Company), L.P. (2018). Recuperado de: www1.hpe.com

[18]. Hyojoon, K., Nick, F., (2013). *Improving network management with software defined networking*. IEEE Communications Magazine, Volume 51, 114-119. Febrero, 2013, De IEEE Xplore Base de datos

[19]. IEEE P802.3ad (2017). Recuperado de: http://www.ieee802.org/3/ad/

[20]. IPERF (2018). Recuperado de: https://iperf.fr/

[21]. ITPRICE (2018). Recuperado de: http://itprice.com/

[22]. ITU-T. (2012). *Focus Group on Cloud Computing Technical Report Part 3: Requirements and Framework Architecture of Cloud Infrastructure*. febrero, 2012, de ITU-T FG Cloud TR.

[23]. ITU-T. (2014). *Framework of software-defined networking*. junio, 2014, de ITU-T Documento de recomendaciones.

[24]. Kreutz, D., Ramos, F., Verissimo, P., Rothenberg, C., Azodolmolky, S. y Uhlig, S. (2015). *Software-Defined Networking: A Comprehensive Survey*. Proceedings of the IEEE, 103, 14-76. Enero, 2017, De IEEE Xplore Base de datos.

[25]. Leao, R., Akira, A., Marie, C. y Rodriguez, L. (2014). *Using Mininet for emulation and prototyping Software-Defined Networks*. COLCOM IEEE Colombian Conference. julio, 2014, De IEEE Xplore Base de datos.

[26]. Luo, J., Pettit, J., Casado, M., Lockwood, J. y McKeown, N. (2007). *Prototyping Fast, Simple, Secure Switches for Ethane*. IEEE Hot Interconnects. agosto, 2007, De IEEE Xplore Base de datos.

[27]. Marek, S. (2016). *AT&T CTO Expects SDN to Reduce OpEx Costs by 40%.* junio, 2016, de SDNxCentral. Recuperado de: https://www.sdxcentral.com/articles/news/att-cto-expects-sdn-reduce-opex-costs-40/2016/06/

[28]. Medved, J., Malachovsky, D., Sebin, J., Kannan, V., Metz, C., Montin, N., Kuzma, D., Jamrich, S., Krnac, Z., Madaminov, A., Vanko, A. y Grewal, B. (2014). *OpenDaylight OpenFlow Manager (OFM) App*. enero, 2014, de Cisco. Recuperado de: https://github.com/CiscoDevNet/OpenDaylight-Openflow-App/blob/master/README.md

[29]. Murat, K., Arjan, D (2017), *Service Cost in Software Defined Networking (SDN)*, 2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA), Marzo, 2017. De IEEE Xplore Base de datos

[30]. Dano, M. (2016). *AT&T CTO on capex: 'It's certainly not going up. It's certainly going down.* Agosto, 2016, de FierceWireless. Recuperado de: https://www.fiercewireless.com/wireless/at-t-cto-capex-it-s-certainly-not-going-up-it-s-certainly-going-down

[31]. Mininet (2017). [Online]. Recuperado de: http://mininet.org/

[32]. Nadeau, T., Gray, K. (2013). *Software Defined Networks*. Estados Unidos: O'Really.

[33]. Newman, P., Edwards, W., Hinden, R., Hoffman, E., Ching Liaw, F., Lyon, T. y Minshall, G. (1996). *Ipsilon's general switch management protocol specification version 1.1*. agosto, 1996, de IETF RFC1987. Recuperado de: https://tools.ietf.org/html/rfc1987

[34]. Openflow (2017). Recuperado de: http://www.openflow.org.

[35]. Open Data Center Alliance. (2014). *Master USAGE MODEL: Software-Defined Networking Rev. 2.0*. 2014, de Open Data Center Alliance. Recuperado de: https://opendatacenteralliance.org/article/software-defined-networking-master-usage-model-rev2-0/

[36]. Open Nerworking Foundation. (2012). *Software-Defined Networking: The New Norm for Networks*. abril, 2012, de Open Nerworking Foundation, white paper.

[37]. Open Nerworking Foundation. (2015). *OpenFlow Switch Specification Version 1.3.5*. marzo, 2015, de Open Nerworking Foundation. Recuperado de: https://3vf60mmveq1g8vzn48q2o71a-wpengine.netdna-ssl.com/wp-content/uploads/2014/10/openflow-switch-v1.3.5.pdf

[38]. Rao, S. (2015). *SDN Series Part Eight: Comparison Of Open Source SDN Controllers*. marzo, 2015, de The New Stack. Recuperado de: https://thenewstack.io/sdn-series-part-eight-comparison-of-open-source-sdn-controllers/.

[39]. Red Telemática UNMSM (2017). [Online]. Recuperado de: http://telematica.unmsm.edu.pe/

[40]. Rekhter, Y., Lie, T. y Huarez, S. (2006). *A Border Gateway Protocol 4 (BGP-4).* enero, 2016, de IETF RFC4271. Recuperado de: https://tools.ietf.org/html/rfc4271

[41]. Rooney, S. y Van Der Merwe, J. (1998). *The Tempest: a framework for safe, resource assured, programmable networks*, IEEE Communications Magazine, 36, pp42-53. 1998, octubre, De IEEE Xplore Base de datos.

[42]. SDN HUB. (2015). *OpenDaylight Application Developer's tutorial*. marzo, 2015, de SDN HUB. Recuperado de: http://sdnhub.org/tutorials/opendaylight/

[43]. SDXCENTRAL. (2015). *Featured Use Case Studies: HP SDN for Education*. setiembre, 2015, de SDXCENTRAL. Recuperado de: https://www.sdxcentral.com/articles/featured/sdn-for-education-hp-use-case/2015/09/

[44]. Shailendra, M. y Mohammed, R. (2017). *Software Defined Networking: Research Issues,Challenges and Opportunities*. India Journal of Science and Technology, Majmaah University.

[45]. Stallings, W. (2015). *Foundations of Modern Networking: SDN, NFV, QoE, IoT, and Cloud*. Estados Unidos: Pearson Eduaction.

[46]. VYOS (2018). [Online]. Recuperado de: https://vyos.io/es/

[47]. Yang, L., Dantu, R., Anderson. T. y Gopal, R. (2014). *Forwarding and Control Element Separation (ForCES) Framework*. abril, 2014, de IETF RFC3746. Recuperado de: http://tools.ietf.org/html/rfc374

ANEXOS

ANEXO 1: MATRIZ DE CONSISTENCIA

Matriz de consistencia				
Título: Propuesta de diseño de una red de datos de área local bajo la arquitectura de redes definidas por software para la red telemática de la Universidad Nacional Mayor de San Marcos				
Problema	**Objetivos**	**Hipótesis**	**Variables**	**Tipo de Investigación**
Hipótesis general de la investigación: ¿Es posible que una red LAN de la Red Telemática de la UNMSM bajo la arquitectura SDN permita optimizar la gestión de los dispositivos de red mediante el control centralizado?	**Objetivo principal de la Investigación:** Diseñar de una red LAN bajo la arquitectura SDN para la Red Telemática de la UNMSM en un entorno de simulación, que permita optimizar la gestión e interoperabilidad de los dispositivos de red mediante un control centralizado.	**Hipótesis general:** H1: La red LAN de la Red Telemática de la UNMSM bajo una arquitectura SDN optimizara la gestión de los dispositivos de red mediante un control centralizado. **Hipótesis Nula:** H0: La red LAN de la Red Telemática de la UNMSM bajo una arquitectura SDN NO optimizara la gestión de los dispositivos de red mediante un control centralizado.	**Dependiente:** - Gestión de los dispositivos de red de la red LAN de la Red Telemática de la UNMSM. - Interoperabilidad entre los distintos dispositivos de la red LAN de la Red Telemática de la UNMSM. **Independiente:** - Diseño de la arquitectura de red SDN en la Red Telemática de la UNMSM.	Tipo de tesis cuasiexperimental. Se tendrá una metodología basada en tres etapas: documentación, evaluación de la información recolectada y diseño. - Documentación - Evaluación de la información obtenida. - Propuesta de diseño - Experimental.
Hipótesis específicas:	**Objetivo Secundario de la Investigación:**	**Hipótesis específicas de la Investigación:**		
Hipótesis especifica N1: ¿Cómo obtengo el desempeño actual de la gestión de los dispositivos de la red LAN de la Red Telemática de la UNMSM?	**Objetivo específico N1:** Analizar el estado de la Red Telemática de la UNMSM según la forma de la gestión de los dispositivos de red en redes de arquitectura tradicionales.	**Hipótesis especifica N1:** El análisis del estado actual de la Red Telemática de la UNMSM bajo una arquitectura tradicional permitirá obtener el desempeña actual de la gestión de los dispositivos de red LAN.		
Hipótesis especifica N2: ¿Cómo se puede optimizar la gestión de los dispositivos de red LAN de la Red Telemática que están bajo una arquitectura de red tradicional?	**Objetivo específico N2:** Diseñar la red LAN de la Red Telemática de la UNMSM bajo una arquitectura de red SDN mediante el programa de simulación Mininet buscando optimizar la gestión de los dispositivos de red.	**Hipótesis especifica N2:** Se optimizará la gestión de los dispositivos de red LAN mediante un diseño de la red LAN de la Red Telemática bajo la arquitectura de red SDN.		

Hipótesis especifica N3: ¿Cuál sería el impacto de mejora de la gestión de los dispositivos de la red LAN de la Red Telemática de la UNMSM bajo una arquitectura de red SDN?	**Objetivo específico N3:** Evaluar la gestión de los dispositivos de red de la arquitectura SDN propuesta para la red LAN de La Red Telemática bajo el entorno simulado.	**Hipótesis especifica N3:** El diseño de la red LAN bajo arquitectura de red SDN impactara de forma positiva la gestión de los dispositivos de red de la red LAN de la Red Telemática de la UNMSM.		

ANEXO 2: SCRIP DE LA TOPOLOGÍA DE LA RED TELEMÁTICA SIMULADA

EL script escrito en Python "redtelematica_run.py" que se ha implementado para ser simulado en Mininet con el fin de simular la topología propuesta en el cápitulo 4 es el siguiente:

```
"""Topologia red telematica - UNMSM

---Red Telematica  - Sede Central---
---Core---
---Distribucion Nodos---
---Acceso---

"""

from mininet.topo import Topo
from mininet.net import Mininet
from mininet.util import dumpNodeConnections

class RedTelematica( Topo ):
    "Topologia"

    def build( self ):
        # host Red Campus
        h1 = self.addHost( 'h1')
        h2 = self.addHost( 'h2')
        h3 = self.addHost( 'h3')
        h4 = self.addHost( 'h4')
        h5 = self.addHost( 'h5')
        h6 = self.addHost( 'h6')
        h7 = self.addHost( 'h7')
        h8 = self.addHost( 'h8')
        h9 = self.addHost( 'h9')

        # host Red Telematica
        h11 = self.addHost( 'h11')
        h12 = self.addHost( 'h12')
        h13 = self.addHost( 'h13')
```

```
# host Sede Central
h10 = self.addHost( 'h10')

# switch Core
s1 = self.addSwitch( 's1' )
s2 = self.addSwitch( 's2' )

# switch nodos distribucion
s3 = self.addSwitch( 's3' )
s4 = self.addSwitch( 's4' )
s5 = self.addSwitch( 's5' )
s6 = self.addSwitch( 's6' )
s7 = self.addSwitch( 's7' )
s41 = self.addSwitch( 's41' )
s42 = self.addSwitch( 's42' )
s43 = self.addSwitch( 's43' )
s44 = self.addSwitch( 's44' )

# switch acceso
# switch nodo1
s17 = self.addSwitch( 's17' )
s18 = self.addSwitch( 's18' )
s19 = self.addSwitch( 's19' )
s8 = self.addSwitch( 's8' )
# switch nodo2
s24 = self.addSwitch( 's24' )
s25 = self.addSwitch( 's25' )
s26 = self.addSwitch( 's26' )
s9 = self.addSwitch( 's9' )
# switch nodo3
s27 = self.addSwitch( 's27' )
s28 = self.addSwitch( 's28' )
s29 = self.addSwitch( 's29' )
s10 = self.addSwitch( 's10' )
# switch nodo4
s30 = self.addSwitch( 's30' )
s31 = self.addSwitch( 's31' )
s32 = self.addSwitch( 's32' )
s33 = self.addSwitch( 's33' )
s11 = self.addSwitch( 's11' )
# switch nodo5
s12 = self.addSwitch( 's12' )
# switch nodo6
s13 = self.addSwitch( 's13' )
# switch nodo7
s14 = self.addSwitch( 's14' )
# switch nodo8
s34 = self.addSwitch( 's34' )
s35 = self.addSwitch( 's35' )
s36 = self.addSwitch( 's36' )
s37 = self.addSwitch( 's37' )
s15 = self.addSwitch( 's15' )
# switch nodo9
s38 = self.addSwitch( 's38' )
s39 = self.addSwitch( 's39' )
```

```
s40 = self.addSwitch( 's40' )
s16 = self.addSwitch( 's16' )


# switch Red Telematica
s20 = self.addSwitch( 's20' )
s21 = self.addSwitch( 's21' )
s22 = self.addSwitch( 's22' )

# switch sede central
s23 = self.addSwitch( 's23' )

# links entre cores
self.addLink( s1, s2)

# links distribucion a core1
self.addLink( s3, s1)
self.addLink( s4, s1)
self.addLink( s5, s1)
self.addLink( s6, s1)
self.addLink( s7, s1)
self.addLink( s41, s1)

# links distribucion a core2
self.addLink( s42, s2)
self.addLink( s43, s2)
self.addLink( s44, s2)


# link acceso
# link nodo1
self.addLink( s17, s3)
self.addLink( s18, s3)
self.addLink( s19, s3)
self.addLink( s8, s3)
# link nodo2
self.addLink( s24, s4)
self.addLink( s25, s4)
self.addLink( s26, s4)
self.addLink( s9, s4)
# link nodo3
self.addLink( s27, s5)
self.addLink( s28, s5)
self.addLink( s29, s5)
self.addLink( s10, s5)
# link nodo4
self.addLink( s30, s6)
self.addLink( s31, s6)
self.addLink( s32, s6)
self.addLink( s33, s6)
self.addLink( s11, s6)
# link nodo5
self.addLink( s12, s7)
# link nodo6
self.addLink( s13, s41)
```

```
        # link nodo7
        self.addLink( s14, s42)
        # link nodo8
        self.addLink( s34, s43)
        self.addLink( s35, s43)
        self.addLink( s36, s43)
        self.addLink( s37, s43)
        self.addLink( s15, s43)
        # link nodo9
        self.addLink( s38, s44)
        self.addLink( s39, s44)
        self.addLink( s40, s44)
        self.addLink( s16, s44)

        # link Red Telematica
        self.addLink( s20, s1)
        self.addLink( s21, s1)
        self.addLink( s22, s1)

        # link Sede Central
        self.addLink( s23, s2)

        # link host acceso
        self.addLink( h1, s8)
        self.addLink( h2, s9)
        self.addLink( h3, s10)
        self.addLink( h4, s11)
        self.addLink( h5, s12)
        self.addLink( h6, s13)
        self.addLink( h7, s14)
        self.addLink( h8, s15)
        self.addLink( h9, s16)

    # Links host Red Telematica
        self.addLink( s20, h11)
        self.addLink( s21, h12)
        self.addLink( s22, h13)

    # Links host Sede Central
        self.addLink( s23, h10)

# Permite al archivo ser importado con `mn --custom <filename> -
-topo minimal`
topos = {
    'RedTelematica': RedTelematica
}
```

ANEXO 3: TABLAS DE FLUJO DEL SWITCH 22 EN OPENDAYLIGHT

El controlador Opendaylight tiene una base de datos de todos los switch y host de la red SDN. Se obtuvo las tablas de flujo instaladas en el switch en el controlador Opendaylight a través del API RESTCONF en código XML:

URL:
http://192.168.122.11:8181/restconf/operational/opendaylight-inventory:nodes/node/openflow:22/table/0/

```
<table xmlns="urn:opendaylight:flow:inventory">
    <id>0</id>
    <flow>
        <id>L2switch-1</id>
        <cookie_mask>0</cookie_mask>
        <priority>10</priority>
        <table_id>0</table_id>
        <flow-statistics
xmlns="urn:opendaylight:flow:statistics">
            <packet-count>21</packet-count>
            <byte-count>1666</byte-count>
            <duration>
                <second>209</second>
                <nanosecond>942000000</nanosecond>
            </duration>
        </flow-statistics>
        <match>
            <ethernet-match>
                <ethernet-destination>
                    <address>00:00:00:00:00:0c</address>
                </ethernet-destination>
                <ethernet-source>
                    <address>00:00:00:00:00:02</address>
                </ethernet-source>
            </ethernet-match>
        </match>
        <flags></flags>
        <idle-timeout>600</idle-timeout>
        <hard-timeout>300</hard-timeout>
        <cookie>30264184949592973313</cookie>
        <instructions>
            <instruction>
                <order>0</order>
                <apply-actions>
                    <action>
                        <order>0</order>
                        <output-action>
                            <output-node-connector>3</output-
node-connector>
                            <max-length>65535</max-length>
                        </output-action>
                    </action>
                </apply-actions>
            </instruction>
        </instructions>
    </flow>
    <flow>
        <id>L2switch-11</id>
        <cookie_mask>0</cookie_mask>
        <priority>10</priority>
```

```
        <table_id>0</table_id>
        <flow-statistics
xmlns="urn:opendaylight:flow:statistics">
            <packet-count>20</packet-count>
            <byte-count>1624</byte-count>
            <duration>
                <second>209</second>
                <nanosecond>695000000</nanosecond>
            </duration>
        </flow-statistics>
        <match>
            <ethernet-match>
                <ethernet-destination>
                    <address>00:00:00:00:00:0c</address>
                </ethernet-destination>
                <ethernet-source>
                    <address>00:00:00:00:00:09</address>
                </ethernet-source>
            </ethernet-match>
        </match>
        <flags></flags>
        <idle-timeout>600</idle-timeout>
        <hard-timeout>300</hard-timeout>
        <cookie>3026418949592973323</cookie>
        <instructions>
            <instruction>
                <order>0</order>
                <apply-actions>
                    <action>
                        <order>0</order>
                        <output-action>
                            <output-node-connector>3</output-
node-connector>
                            <max-length>65535</max-length>
                        </output-action>
                    </action>
                </apply-actions>
            </instruction>
        </instructions>
    </flow>
    <flow>
        <id>L2switch-8</id>
        <cookie_mask>0</cookie_mask>
        <priority>10</priority>
        <table_id>0</table_id>
        <flow-statistics
xmlns="urn:opendaylight:flow:statistics">
            <packet-count>20</packet-count>
            <byte-count>1624</byte-count>
            <duration>
                <second>209</second>
                <nanosecond>782000000</nanosecond>
            </duration>
        </flow-statistics>
        <match>
```

```
            <ethernet-match>
                <ethernet-destination>
                    <address>00:00:00:00:00:08</address>
                </ethernet-destination>
                <ethernet-source>
                    <address>00:00:00:00:00:0c</address>
                </ethernet-source>
            </ethernet-match>
        </match>
        <flags></flags>
        <idle-timeout>600</idle-timeout>
        <hard-timeout>300</hard-timeout>
        <cookie>3026418949592973320</cookie>
        <instructions>
            <instruction>
                <order>0</order>
                <apply-actions>
                    <action>
                        <order>0</order>
                        <output-action>
                            <output-node-connector>1</output-
node-connector>
                            <max-length>65535</max-length>
                        </output-action>
                    </action>
                </apply-actions>
            </instruction>
        </instructions>
    </flow>
    <flow-table-statistics
xmlns="urn:opendaylight:flow:table:statistics">
        <active-flows>24</active-flows>
        <packets-matched>451476</packets-matched>
        <packets-looked-up>454419</packets-looked-up>
    </flow-table-statistics>
</table>
```

ANEXO 4: CONFIGURACIÓN BGP – MODÚLO BGP-PCEP

La configuración se realizó en código XML según Opendaylight, el scrip utilizado fue el siguiente:

```
URL: /restconf/config/openconfig-network-instance:network-
instances/network-instance/global-bgp/openconfig-network-
instance:protocols


<protocol xmlns="http://openconfig.net/yang/network-instance">
    <name>bgp-example</name>
    <identifier xmlns:x="http://openconfig.net/yang/policy-
types">x:BGP</identifier>
    <bgp
xmlns="urn:opendaylight:params:xml:ns:yang:bgp:openconfig-
extensions">
```

```
        <global>
            <config>
                <router-id>192.168.122.11</router-id>
                <as>65100</as>
            </config>
        </global>
    </bgp>
</protocol>
```

```
URL: http://192.168.122.11:8181/restconf/config/openconfig-
network-instance:network-instances/network-instance/global-
bgp/openconfig-network-instance:protocols/protocol/openconfig-
policy-types:BGP/bgp-example/bgp/neighbors

<neighbor
xmlns="urn:opendaylight:params:xml:ns:yang:bgp:openconfig-
extensions">
    <neighbor-address>192.168.122.250</neighbor-address>
    <config>
        <route-flap-damping>false</route-flap-damping>
        <peer-as>65100</peer-as>
        <peer-type>INTERNAL</peer-type>
        <send-community>NONE</send-community>
    </config>
    <route-reflector>
        <config>
            <route-reflector-client>false</route-reflector-
client>
        </config>
    </route-reflector>
    <timers>
        <config>
            <keepalive-interval>60</keepalive-interval>
            <hold-time>180</hold-time>
            <connect-retry>10</connect-retry>
        </config>
    </timers>
    <transport>
        <config>
            <mtu-discovery>false</mtu-discovery>
            <remote-port>179</remote-port>
            <passive-mode>false</passive-mode>
        </config>
    </transport>
    <afi-safis>
        <afi-safi>
             <afi-safi-name
xmlns:x="http://openconfig.net/yang/bgp-types">x:IPV4-
UNICAST</afi-safi-name>
        </afi-safi>
    </afi-safis>
</neighbor>
```

```
URL: http://192.168.122.11:8181/restconf/config/odl-bgp-peer-
acceptor-config:bgp-peer-acceptor-config/default

<bgp-peer-acceptor-config
xmlns="urn:opendaylight:params:xml:ns:yang:odl-bgp-peer-
acceptor-config">
    <config-name>default</config-name>
    <binding-address>0.0.0.0</binding-address>
    <binding-port>179</binding-port>
</bgp-peer-acceptor-config>
```

```
URL: http://192.168.122.11:8181/restconf/config/openconfig-
network-instance:network-instances/network-instance/global-
bgp/openconfig-network-instance:protocols/protocol/openconfig-
policy-types:BGP/bgp-example/bgp/neighbors

<neighbor
xmlns="urn:opendaylight:params:xml:ns:yang:bgp:openconfig-
extensions">
    <neighbor-address>10.25.25.10</neighbor-address>
    <config>
        <peer-group>application-peers</peer-group>
    </config>
</neighbor>
```

```
URL: http://192.168.122.11:8181/restconf/config/bgp-
rib:application-rib/10.25.25.10/tables/bgp-types:ipv4-address-
family/bgp-types:unicast-subsequent-address-family/bgp-
inet:ipv4-routes

<ipv4-route xmlns="urn:opendaylight:params:xml:ns:yang:bgp-
inet">
    <path-id>0</path-id>
    <prefix>10.0.0.11/32</prefix>
    <attributes>
        <as-path></as-path>
        <origin>
            <value>igp</value>
        </origin>
        <local-pref>
            <pref>100</pref>
        </local-pref>
        <ipv4-next-hop>
            <global>10.11.1.1</global>
        </ipv4-next-hop>
    </attributes>
</ipv4-route>
```